Proceedings of the 14th Sound and Music Computing Conference 2017

SMC 2017
July 5-8, 2017
Espoo, Finland
Edited by Tapio Lokki, Jukka Pätynen, and Vesa Välimäki

GENELEC®

NATIVE INSTRUMENTS

Akustinen Seura ry
Acoustical Society of Finland

applied sciences
an open access journal by MDPI

Aalto University

Aalto University
School of Science
School of Electrical Engineering
School of Arts, Design, and Architecture

Published by Aalto University

ISBN 978-952-60-3729-5
ISSN 2518-3672

http://smc2017.aalto.fi

Table of Contents

Keynotes

Posters

Music performance analysis and rendering

Social interaction

Music information retrieval

Spatial sound and sonification

Computer music languages and software

Analysis, synthesis, and modification of sound

Computer-based music analysis

Automatic systems and interactive performance

Preface

We are proud to present the 14th edition of the Sound and Music Computing Conference (SMC-17). The conference is hosted by Aalto University in Otaniemi, Espoo, Finland.

This year's theme "Wave in time and space" contains a language joke, as the Finnish word "aalto" appearing in the name of our university means "a wave". When the Aalto University was formed in 2010 by merging three Finnish universities, Helsinki University of Technology, Helsinki Business School, and the University of Arts and Design, the SMC-related activities were distributed in three different units, the Department of Signal Processing and Acoustics, the Department of Media Technology (currently Computer Science), and the Department of Media. In spring 2017, the Aalto Acoustics Lab was founded as a research center gathering all SMC-related researchers of Aalto University. We have our own office building and acoustic measurement laboratories next to each other in the main Aalto campus in Otaniemi, Espoo. The Acoustics Lab was in fact the old name of the acoustics research unit in the technical university, but it was abandoned about ten years ago, when the university structure was re-organized. It was now the right time to start using it again. The SMC-17 conference is the first common activity of the new Aalto Acoustics Lab: the chair Vesa Välimäki, the papers chair Tapio Lokki, and the music chair Koray Tahiroğlu represent the three different units, which meet at the Aalto Acoustics Lab.

The scientific program of SMC-17 includes 3 keynote lectures, 45 oral presentations, and 20 posters. In addition, there are 15 sound installations or fixed media, and 14 music pieces to be presented in three concerts. Furthermore, a demo session at the Aalto Acoustics Lab is open to all SMC-17 goers. The 65 papers published in these proceedings were selected out of 85 submissions by the program committee. We are honored to welcome keynote speakers Dr. Toshifumi Kunimoto, a famous synthesizer designer from Yamaha (Hamamatsu, Japan), Prof. Anssi Klapuri, the CTO of Yousician (Helsinki, Finland), a company producing musical games, and Dr. Simon Waters (Belfast, UK), a reputable composer of electroacoustic music.

The SMC conference series has been running without constant financial support from any association for many years. We are always getting many participants, who sponsor this conference series by paying the registration fee, which can be kept relatively low in comparison to some other international conferences. With the help of external sponsors, it is possible to not only offer access to all oral and poster presentations, keynote talks, and demos, but also offer free lunches, coffee breaks, concerts, and a banquet to all SMC-17 participants. We would like to express our gratitude to our sponsors: Genelec, Native Instruments, Applied Sciences – Basel, the Acoustical Society of Finland, and the Federation of Finnish Learned Societies.

We welcome you all to Finland and to Aalto University, which you may remember as the wave university. We hope that you will enjoy SMC-17!

Otaniemi, Espoo, June 19, 2017

Vesa Välimäki SMC-17 Chair	Tapio Lokki SMC-17 Papers Chair

Committee

Chair	Prof. Vesa Välimäki
Papers Chair	Prof. Tapio Lokki
Music Chair	Dr. Koray Tahiroğlu
Treasurer	Dr. Jukka Pätynen
Web Manager	Mr. Fabian Esqueda
Practicalities	Ms. Lea Söderman Ms. Tuula Mylläri

Sponsors

GENELEC®

applied sciences

NATIVE INSTRUMENTS

Akustinen Seura ry
Acoustical Society of Finland

Aalto University

Reviewers

Paulo Antonio Andrade Esquef
Federico Avanzini
Stephen Barrass
Emmanouil Benetos
Roberto Bresin
Carl Bussey
Emilios Cambouropoulos
Sergio Canazza
Chris Chafe
John Dack
Matthew Davies
Giovanni De Poli
Stefano Delle Monache
Myriam Desainte-Catherine
Cumhur Erkut
Arthur Flexer
Dominique Fober
Federico Fontana
Anders Friberg
Daniel Gaertner
Woon-Seng Gan
Bruno Giordano
Werner Goebl
Huseyin Hacihabiboglu
Pierre Hanna
Andrew Horner
Daniel Hug
Pierre Jouvelot
Hanna Järveläinen
Haruhiro Katayose
Peter Knees
Tapio Lokki
Raphael Marczak
Filipe Cunha Monteiro Lopes
Hanna Lukashevich
Sylvain Marchand
Matija Marolt
Luis Gustavo Martins
Matthias Mauch
Davide Andrea Mauro
Annamaria Mesaros
Juhan Nam
Felipe Orduña
Sebastian Paquette
Rui Pedro Paiva
Julian Parker
Sandra Pauletto
Alfonso Perez
Pedro J. Ponce De León
Ville Pulkki
Jukka Pätynen
Rudolf Rabenstein
Marc Recasens
Davide Rocchesso
Antonio Roda
Gerard Roma
Diemo Schwarz
Maximilian Schäfer
Véronique Sebastien
Stefania Serafin
Malcolm Slaney
Julius Smith
Simone Spagnol
Tapio Takala
Luis Teixeira
George Tzanetakis
Domenico Vicinanza
Tuomas Virtanen
Gualtiero Volpe
Vesa Välimäki
Hsin-Min Wang
Stefan Weinzierl
Kazuyoshi Yoshii
Paolo Zavagna

Special Issue, Call for Papers

Sound and music computing is a young and highly multidisciplinary research field. It combines scientific, technological, and artistic methods to produce, model, and understand audio and sonic arts with the help of computers. Sound and music computing borrows methods, for example, from computer science, electrical engineering, mathematics, musicology, and psychology.

In this Special Issue, we want to address recent advances in the following topics:

- Analysis, synthesis, and modification of sound
- Automatic composition, accompaniment, and improvisation
- Computational musicology and mathematical music theory
- Computer-based music analysis
- Computer music languages and software
- High-performance computing for audio
- Interactive performance systems and new interfaces
- Multi-modal perception and emotion
- Music information retrieval
- Music games and educational tools
- Music performance analysis and rendering
- Robotics and music
- Room acoustics modeling and auralization
- Social interaction in sound and music computing
- Sonic interaction design
- Sonification
- Soundscapes and environmental arts
- Spatial sound
- Virtual reality applications and technologies for sound and music

Submissions are invited for both original research and review articles. Additionally, invited papers based on excellent contributions to recent conferences in this field will be included in this Special Issue; for example, from the 2017 Sound and Music Computing Conference SMC-17. We hope that this collection of papers will serve as an inspiration for those interested in sound and music computing.

Manuscript Submission Information

Manuscripts should be submitted online at www.mdpi.com by registering and logging in to this website. Once you are registered, click here to go to the submission form. Manuscripts can be submitted until the deadline. All papers will be peer-reviewed. Accepted papers will be published continuously in the journal (as soon as accepted) and will be listed together on the special issue website. Research articles, review articles as well as short communications are invited. For planned papers, a title and short abstract (about 100 words) can be sent to the Editorial Office for announcement on this website.

Submitted manuscripts should not have been published previously, nor be under consideration for publication elsewhere (except conference proceedings papers). All manuscripts are thoroughly refereed through a single-blind peer-review process. A guide for authors and other relevant information for submission of manuscripts is available on the Instructions for Authors page. *Applied Sciences* is an international peer-reviewed open access monthly journal published by MDPI.

Please visit the Instructions for Authors page before submitting a manuscript. The Article Processing Charge (APC) for publication in this open access journal is 1200 CHF (Swiss Francs). Submitted papers should be well formatted and use good English. Authors may use MDPI's English editing service prior to publication or during author revisions.

Guest Editors

Prof. Dr. Tapio Lokki
Prof. Dr. Stefania Serafin
Prof. Dr. Meinard Müller
Prof. Dr. Vesa Välimäki

Keynote 1:

Toshifumi Kunimoto (Yamaha, Japan)

History of Yamaha's electronic music synthesis

This keynote presentation will cover Yamaha's history as an electronic keyboard instrument manufacturer. Yamaha entered the electronic organ business in 1959 with the release of the Electone D-1. In the beginning, Yamaha's organs were designed using analog circuitry. Nowadays, Yamaha uses state-of-the-art digital signal processing (DSP) technologies in their products. This knowledge of DSP technologies for audio and musical applications can be applied not only to musical instruments, but also to audio equipment. This presentation will discuss the issues that arose during Yamaha's transition from analog to digital, as well as introduce some of the latest products Yamaha has developed with their cutting-edge DSP technologies.

About the speaker: Toshifumi Kunimoto was born in Sapporo, Hokkaido, Japan in 1957. He received the B.S., M.S. and Ph.D. degrees for his work on ARMA digital filter design from the Faculty of Engineering, Hokkaido University, Sapporo, Japan in 1980, 1982, and 2017, respectively. In 1982 Toshifumi Kunimoto joined Yamaha, where he has designed large-scale integration (LSI) systems for numerous musical instruments such as the Electone, Yamaha's trademark electronic organ line. He has created numerous signal processing technologies used in Yamaha synthesizers and pro-audio equipment. Among his designs are the famous Yamaha VL1 Virtual Acoustic Synthesizer and the CP1 electronic piano. He has also contributed to several Japanese specialized music and audio publications with articles describing the behavior of analog audio equipment such as guitar stompboxes. Since 2008, he has worked at Yamaha's Center for Research & Development in Hamamatsu, Japan.

Keynote 2:

Simon Waters (Queen's University Belfast, UK)

Revisiting the ecosystemic approach

It is ten years since the publication of my "Performance Ecosystems: Ecological approaches to musical interaction" in the EMS07 Proceedings, a paper in which I attempted to think through the implications of regarding player, instrument and environment as contiguous and interpenetrating in musical activity. As I acknowledged at the time this drew on the work of my colleague John Bowers and on that of my community of research students at the University of East Anglia. In the intervening period the studios and research centre there have been closed down, and the utility of thinking in such a manner has been tested by my moving to the (historically) more "interaction design" focused context of the Sonic Arts Research Centre in Belfast.

Speaking at SMC 2017 affords me the opportunity to reflect both on antecedents to the ideas in the original paper, and on the implications of ecosystemic thought *beyond* musical performance, in the related disciplines of design and computing, and in human conduct more generally. As my musical concerns have during the same period increasingly focused on improvisation, this also leads to a consideration of time, timing and timescales in such conduct, and of the immense resource represented by surviving acoustic musical instruments - which embody histories and ideas which extend their capacity beyond mere *instrumentality*.

<ins>About the speaker:</ins> Dr. Simon Waters joined the staff of the Sonic Arts Research Centre at Queen's University Belfast in September 2012, moving from his previous role as Director of Studios at the University of East Anglia (1994 - 2012). He initially established a reputation as an electroacoustic composer, with awards and commissions and broadcasts in the UK and abroad, working with many contemporary dance and physical theatre companies and visual artists including Ballet Rambert, Adventures in Motion Pictures and the Royal Opera House Garden Venture, and with pioneering multimedia theatre practitioners Moving Being.

From a focus on electroacoustic works for contemporary dance in the 1980s, his work has shifted from studio-based acousmatic composition to a position which reflects his sense that music is at least as concerned with human action as with acoustic fact. His current research explores continuities between tactility and hearing, and is particularly concerned with space as informed by and informing human presence, rather than as a benign "parameter". He has supervised over fifty research students in areas as diverse as improvising machines, inexpertise and marginal competence in musical interaction, silence and silencedness, soundscape and acoustic ecology, real-time audio-visual composition and performance systems design as well as in more explicitly "compositional" areas.

His publications are similarly diverse, recent examples exploring empathy and ethics in improvised conduct, early nineteenth century woodwind instrument production in London, the notion of the "local" in a ubiquitously digitized world, and hybrid physical/virtual instrument design.

Keynote 3:

Anssi Klapuri (Yousician, Finland)

Learnings from developing the recognition engine for a musical instrument learning application

Yousician develops an educational platform that helps our users to learn to play a musical instrument. In a typical setting, the user has a mobile device in front of her and plays a real acoustic instrument to interact with the application running on the device. The application listens to the user's performance via the device's built-in microphone and gives real-time feedback about the performance, while also showing written music on the screen and playing an accompanying backing track. In this talk, I will discuss the learnings that we had while building the music recognition engine for our application. There are several technical challenges in using a musical instrument as a "game controller", particularly so in a cross-platform environment with a very heterogeneous set of devices. The most obvious challenge is to recognise the notes that the user is playing, as the quality and acoustic characteristics of instruments vary widely. The noise conditions for the recognition are often quite demanding: the accompanying music from the device speakers tends to leak back to the microphone and correlates strongly with what the user is supposed to play. At the same time, the application should be robust to background noises such as someone talking in the background and potentially other students playing a few meters away in a classroom setting. Timing and synchronisation sets another challenge: giving feedback about performance timing requires synchronisation accuracy down to roughly 10 ms, whereas the audio I/O latency uncertainties are often an order of magnitude larger on an unknown device. I will discuss some of our solutions to these challenges and our workflow for developing and testing the recognition DSP. I will also discuss briefly the audio processing architecture that we use and the reasons why we decided to build our own audio engine (on top of JUCE) to be in full control of the audio capabilities. I will end with a more personal note as a person who moved from the academia to a startup roughly six years ago, reflecting on the most satisfying vs. frustrating moments and the most interesting or surprising aspects during that time.

About the speaker: Anssi Klapuri received his Ph.D. degree from Tampere University of Technology (TUT), Finland in 2004. He visited as a post-doc researcher at École Centrale de Lille, France, and Cambridge University, UK, in 2005 and 2006, respectively. He led the Audio Research Group at TUT until 2009 and then worked as a Lecturer in the Centre for Digital Music at Queen Mary University of London in 2010–2011. Since 2011, he has worked as the CTO of Yousician, developing educational applications for learning to play musical instruments. His research interests include audio signal processing, auditory modeling, and machine learning.

THE DESIGN OF A LIGHTWEIGHT DSP PROGRAMMING LIBRARY

Victor Lazzarini
Maynooth University,
Ireland
victor.lazzarini@nuim.ie

ABSTRACT

This paper discusses the processes involved in designing and implementing an object-oriented library for audio signal processing in C++ (ISO/IEC C++14). The introduction presents the background and motivation for the project, which is related to providing a platform for the study and research of algorithms, with an added benefit of having an efficient and easy-to-deploy library of classes for application development. The design goals and directions are explored next, focusing on the principles of stateful representations of algorithms, abstraction/ encapsulation, code reuse and connectivity. The paper provides a general walkthrough the current classes and a detailed discussion of two algorithm implementations. Completing the discussion, an example program is presented.

1. INTRODUCTION

In 1998, I introduced version 1.0 of the SndObj library [1], which was at the time most likely one of the first generally available free and open-source general-purpose C++ class libraries for audio processing. It came out at around the same time as another early C++ toolkit, STK [2], which was mostly oriented to synthesis with physical models. The SndObj library included not only signal processing classes but also support for cross-platform realtime audio and MIDI, as it aimed to encompass all the general-purpose audio use cases [3]. The original code was completely based on pre-standard C++, as it appeared around the same time the C++98 was about to be published. It was heavily modelled on Stroustrup's prescriptions [4].

Over time, the library developed to include some of C++98 and later C++03 ideas, but its development suffered from the twists and turns of the language as it began to be standardised. In hindsight, writing thousands of lines in hundreds of source-code files based on a moving target as was C++ twenty years ago was an unwise and perilous task. However, someone had to do it. In the following decade, many C++ object-oriented library projects got developed, as the terrain got firmed underneath them.

My own attention got diverted elsewhere and from about 2005, very little was done to the SndObj library apart from adding wrappers to it and employing it as the basis for Python projects. The code, however, proved to be a good resource for teaching and I extracted many components from it as free-standing functions that I would use in classroom, creating a small function library for my students [5]. This allows the algorithms to be studied and played with, but it is not robust enough to be used more generally.

With the advent of C++11 [6], I have renewed my interest in the language and have realised that it is now in a much more stable state, which would warrant some time and resource investment. For those who work mostly in C, like myself, it is still a system of elephantine proportions that lacks some of the finesse of small languages. It has been described as having "the power, the elegance, and the simplicity of a hand grenade" [1] . However, when used in moderation, it can provide a number of benefits as an extension of C.

This is the background to the development of AuLib [7]. The motivation is to provide a simple, lightweight platform for the study, teaching, and research of digital signal processing (DSP) algorithms for audio, taking advantage of the newer, more established, C++ standards. It also has the added benefit of providing efficient and portable code that can be easily packaged and deployed in general-purpose applications. The fundamental aim is to provide wrappers for algorithms, where the audio processing code can be easily accessed, studied, and modified. This is wrapped in a very thin interface layer that allows easy connectivity of objects and a class hierarchy that emphasises common components and code re-use. The library code attempts to adhere strictly to C++14 [8] standards and best practices, as it aims to provide an example of robust software design.

This paper is organised as follows. We will present the library design and discuss the decisions taken in its development. A tour of the library and its current constituent classes is shown next, exploring it from the perspective of signal generation, processing, and input and output. Finally, two classes are singled out for a more detail discussion and a full program example is presented.

2. LIBRARY DESIGN

The library design borrows from a number of sources, which have shown the best practice in the implementation of audio programming code. One of the guiding aspects was to allow a good deal of flexibility in the construction of classes, instead of mandating the presence of specific components via an abstract base class with a number of empty

[1] Quote attributed to Kenneth C. Dyke, 05/04/97.

virtual methods. Instead, processing methods may or may not exist in a derived class, depending on what they are supposed to implement. They can be given any name, although the established informal nomenclature is to call them `process()`.

The principal base class in the library is `AudioBase`. This is subclassed to implement all different audio-handling objects, from synthesis/processing to signal buffers, function tables, delay lines, and audio IO. The layout of `AudioBase` is informed by a number of basic design decisions that underpin the principles of AuLib.

2.1 Stateful vs. stateless representations

One of the basic motivations for AuLib was to place self-standing algorithms, previously implemented as free functions, within a wrapping object allowing the safe-keeping of internal states. Let's explore this idea with a simple example of a sinusoidal oscillator, described by eq. 1.

$$s(t) = a(t) \sin\left(2\pi \int f(t)\right) \tag{1}$$

A C implementation of such function would have to take account of the sample-by-sample phase values that are produced by the integration of the time-varying frequency $f(t)$. Typically, in a sane implementation, the current value of the phase would be kept externally to the function, and modified as a side-effect (listing 1).

Listing 1. C function implementing eq. 1

```
const double twopi = 8. * atan(1.);
double sineosc(double a, double f,
        double *ph, double sr){
    double s = a * sin(*ph);
    *ph += twopi * f / sr;
    return s;
}
```

While this is entirely appropriate to demonstrate and expose the oscillator algorithm for study, it is clearly not robust enough to be incorporated into a library. Quite rightly, users would expect to be able to use such functions to implement multiple oscillators, in banks, or for amplitude or frequency modulation. In this context, a programmer could inadvertently supply a single phase address to a series of calls to such functions when implementing a bank of oscillators and would clearly fail to get the intended result. While it could work when carefully employed, such as stateless presentation of the algorithm is clearly incomplete.

While there are ways of describing a sine wave oscillator in a stateless or purely functional fashion, once we are committed to define the computation in a stateful form, we need to provide a means to keep an account of the current state. Clearly, a self-contained oscillator will need to maintain the last computed value of the phase, as the algorithm contains an integration. For this, we can wrap the whole algorithm in a class that models its state and the means to get an output sample. A minimal C++ class implementing such oscillator is shown in listing 2.

Listing 2. C++ class implementing eq. 1

```
struct SineOsc {
 double m_ph;
 double m_sr;

 SineOsc(double ph, double sr)
       m_ph(ph), m_sr(sr) {};

 double process(double a, double f){
    double s = a * sin(m_ph);
    m_ph += twopi* f / m_sr;
    return s;
 }
};
```

With an object-oriented implementation, the stateful description of the algorithm is complete and provides enough robustness for use in a variety of contexts. Likewise, if we look across the various types of DSP operations that a library would hope to implement, we will see all sorts of state variables involved. This provides enough motivation for the wrapping of such algorithms in C++ classes.

2.2 Abstraction and encapsulation

In fact, by clearly describing an algorithm as having a state and a means of computing its output, we are abstracting the DSP object as a specific data type. This encapsulates all the kinds of operations we would expect to be able to apply to such an object. What are the things we would like any DSP algorithm to contain? It would be useful for instance for it to hold its output so that we only need to compute it once. Basic attributes such as the sampling rate and the frame size (number of channels in an interleaved signal) would also be essential.

For efficient implementation, the output should not be limited to sample-by-sample computation (as in the minimal example of the oscillator in listing 2). Typically, we would want a block of frames to be generated for each call of a processing method, which may vary in size. A means of registering whether the object is in an error state would also be useful for program diagnostics. In this formulation, a class that models a generic Audio DSP object would contain the following attributes (listing 3)

Listing 3. Attributes of the Audio DSP base class

```
class AudioBase {
 protected:
  uint32_t m_nchnls;// no of channels
  uint32_t m_vframes;// vector size
  std::vector<double> m_vector;
  double m_sr;// sampling rate
  uint32_t m_error;// error code
    ...
};
```

These are protected so that no unintended modification is allowed. This class is for all practical purposes a wrapper around an audio vector (of double floating-point samples).

Methods for basic manipulation are also added: scale, offset, modulation, mixing, and sample access are provided through overloaded operators. Setting and getting samples off the vector are also provided (single channel samples, full blocks, etc.), and to modify the vector size, as well as methods to get the value of the object attributes.

2.3 Code re-use

Since we have embraced, for good reasons, the object-oriented approach, it is very useful to take advantage of inheritance, as well as composition. For this reason, I have designed the class hierarchy from the most general to the most specific, although overall the tree is not very deep (six levels at most). We will see two specific cases in more detail in section 4, but as an example, the `ResonZ` class shows how the re-use of code can be employed. In fig. 1, we see that it is subclassed from a series of parents.

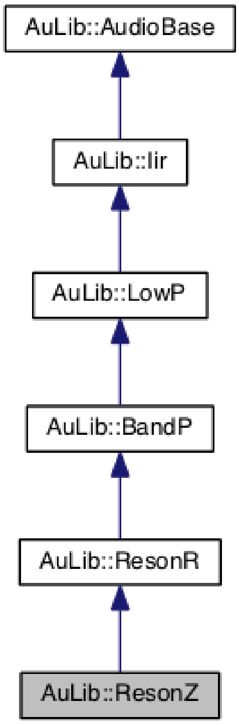


Figure 1. The `ResonZ` class and its parents.

At the top-level, `Iir` implements the basic second-order infinite impulse response (IIR) filter engine in direct form II (eq.2), with externally-defined coefficients.

$$\begin{aligned} w(t) &= x(t) - b_1 w(t-1) - b_2 w(t-2) \\ y(t) &= a_1 w(t) + a_1 w(t-1) + a_2 w(t-2) \end{aligned} \tag{2}$$

The `LowP` class holds a frequency parameter to calculate Butterworth low-pass coefficients; `BandP` adds a bandwidth attribute and re-implements the calculation of coefficients for a Butterworth band-pass configuration; `ResonR` re-implements the coefficient computation for a Resonator with an extra zero at R; `ResonZ` just sets the a_2 coefficient to -1, otherwise using the coefficient update code from its parent.

This shows an example of how each subclass represents mostly small modification of its parent, with most of its code re-used. Another benefit is that if a modification needs to be made (e.g. a bug fix), it does not need to be reproduced at several places (which opens the door for introducing small errors at these different locations).

Code re-use through composition is also employed throughout the library. For example, the `Delay` class holds an `AudioBase` object that implements its delay line, using the inlined access methods provided in that class. The `Balance` class, which implements envelope following and signal amplitude balancing, is made up of two `Rms` objects that are used to measure the RMS amplitude of input signals. `Rms` itself is a specialisation of a first-order low-pass filter class. In another example, the `TableSet` class, which is an utility class for the band-limited oscillator class `BlOsc` is made up of a vector of `FourierTable` objects containing waveform tables.

2.4 Connectivity

Some special attention has been given to the ways in which objects can be easily connected with each other and with code from other libraries (both in C and C++). For this to be achieved, there are two ways in which input and output to objects can be achieved:

1. Through direct pointers to data: these are presented in the form of **`const double*`** to allow signals from other libraries and non-AuLib sources to be inserted as inputs to processes. These, in turn, also return a **`const double*`** to the object vector so that they can be sent to other destinations. This type of connectivity is unsafe, from a C++ perspective, as it requires the programmer to carefully set the vector boundaries, although it is common place within a C-language context.

2. Object references: processing methods also allow connections to and from **`const`** `AudioBase &` variables, which provides more safety since vector boundaries are checked before access. They are the preferred way to pass signals from one library object to another. For convenience, classes overload the **`operator`**`()` as a shortcut for processing methods using object references.

While there is no mandatory way in which this is enforced in derived classes, the informal convention is to provide two processing methods as part of the public interface: one which deals with data pointers (producing and/or consuming arrays) and another that uses object references as input and/or output, as shown in listing 4. These methods delegate to a private virtual DSP method, which does the actual processing for the object and can be overriden in a derived class. This approach allows for good separation of concerns between interface and implementation.

Listing 4. Processing methods and their connectivity

```
class Proc : public AudioBase {
 virtual const double *
    dsp(const double *sig);
 public:
  const double *
    process(const double *sig) {
      return dsp(sig);
    }
  const Proc &
    process(const AudioBase &obj) {
      if(obj.vframes() == m_vframes &&
         obj.nchnls() == m_nchnls) {
          process(obj.vector());
       } else m_error = AULIB_ERROR;
       return *this;
    }
  const Proc &
   operator()(const AudioBase &obj) {
    return process(obj);
   }
 ...
 };
```

No blocking operations (and/or resource allocation) should take place in a processing method. This is also the case for all inline vector manipulation methods (operators, etc.) provided by the base class, which are all real-time safe.

3. A TOUR OF THE LIBRARY

There are currently 61 classes in the library, of which 56 sit in the main AudioBase tree. Fig.2 shows the AuLib:: AudioBase class hierarchy. These classes can be loosely categorised as processing (signal generators and processors, ie. they implement process() methods), function tables, holding mostly constant buffers, and input/output, which allow some form of audio IO through read() or write methods. In addition to these, the library also features note, instrument and score model classes.

3.1 Signal generators

Signal generators in AuLib include standard table-lookup oscillators (discussed in more detail in section 4), sampled-sound and band-limited waveform oscillators, phase generator, table readers, and envelope generators. The Sample Player class takes a buffer/function table containing recorded samples and plays it back with pitch and amplitude control either in a loop or as a single-shot performance. It can handle multichannel sample tables producing multichannel output and uses linear interpolation for table lookup.

The library supports a number of function table classes, derived from FuncTable, which hold waveforms, envelopes, or signal samples. These can be read by oscillators or by table lookup objects, whose indices can be derived from any signal. A phase generator connected to a table reader implements an oscillator algorithm.

The BlOsc class implements band-limited waveform synthesis using wavetables stored in a TableSet object. This will contain a set of band-limited tables that are selected according to the desired fundamental frequency. Currently, TableSet supports classic waveforms (such as Sawtooth, Square and Triangle) constructed using FourierTable objects. However the mechanism can be expanded to handle generalised band-limited waveforms.

The library contains single-segment linear and exponential signal generators, which can be triggered and reset. Extending these, a generalised multi-segment plus release envelope class Envel is provided. It uses a utility class, Segments, that is used to set up a segment list that can be shared among several envelopes (and also used for envelope tables). A pre-packaged four-segment envelope, ADSR (attack-decay-sustain-released) is derived from it as a convenient way to create simple envelopes. The release segment in these classes is triggered by a specific method (release()), which makes the envelope jump immediately to that stage.

3.2 Signal processors

The library contains a basic set of signal processing classes. Seven types of second-order, plus two first-order (low- and high-pass) filters are present, alongside root-mean-square detection and signal balancing. The Delay class implements fixed or variable delays (depending on the choice of overloaded processing functions), with or without feedback. It can implement comb filters, flangers, vibrato and chorus effects. Derived from it, we have a high-order allpass filter and a general-purpose finite impulse response filter (implementing direct convolution, which is discussed in section 4). Delay objects can be tapped by Tap (truncating) or Tapi (interpolating) processors.

Some signal-processing utilities are present. A channel extractor, Chn, takes an interleaved multi-channel input and outputs a requested channel. A signal bus, SigBus, can be used as a mixer, with scaling and offset. Completing these, there is an equal-power panning class, Pan, that produces a stereo output from a mono input signal.

In addition to these time-domain processing classes, AuLib provides support for streaming spectral processing using the short-time Fourier transform and its derivative, the phase vocoder. Free (stateless) functions for complex and real input discrete Fourier transform (using a radix-2 algorithm) are implemented from first principles and are also available for general use. A partitioned convolution class is also implemented using these functions.

3.3 Input and output

An asynchronous (non-blocking) input and output (IO) facility is provided as part of the library, through the SoundIn and SoundOut classes. The interface is fairly agnostic as far as its implementation is concerned. Currently, it provides a frontend to libsndfile[2], for soundfile IO, portaudio[3], for realtime device IO, and std::iostream for standard text IO.

Users of the library do not actually depend on these two IO classes. For instance, an application could place the processing classes directly in an audio system callback (e.g. through Jack[4]), without the use of any AuLib IO object.

[2] http://www.mega-nerd.com/libsndfile

[3] http://www.portaudio.com

[4] http://jackaudio.org

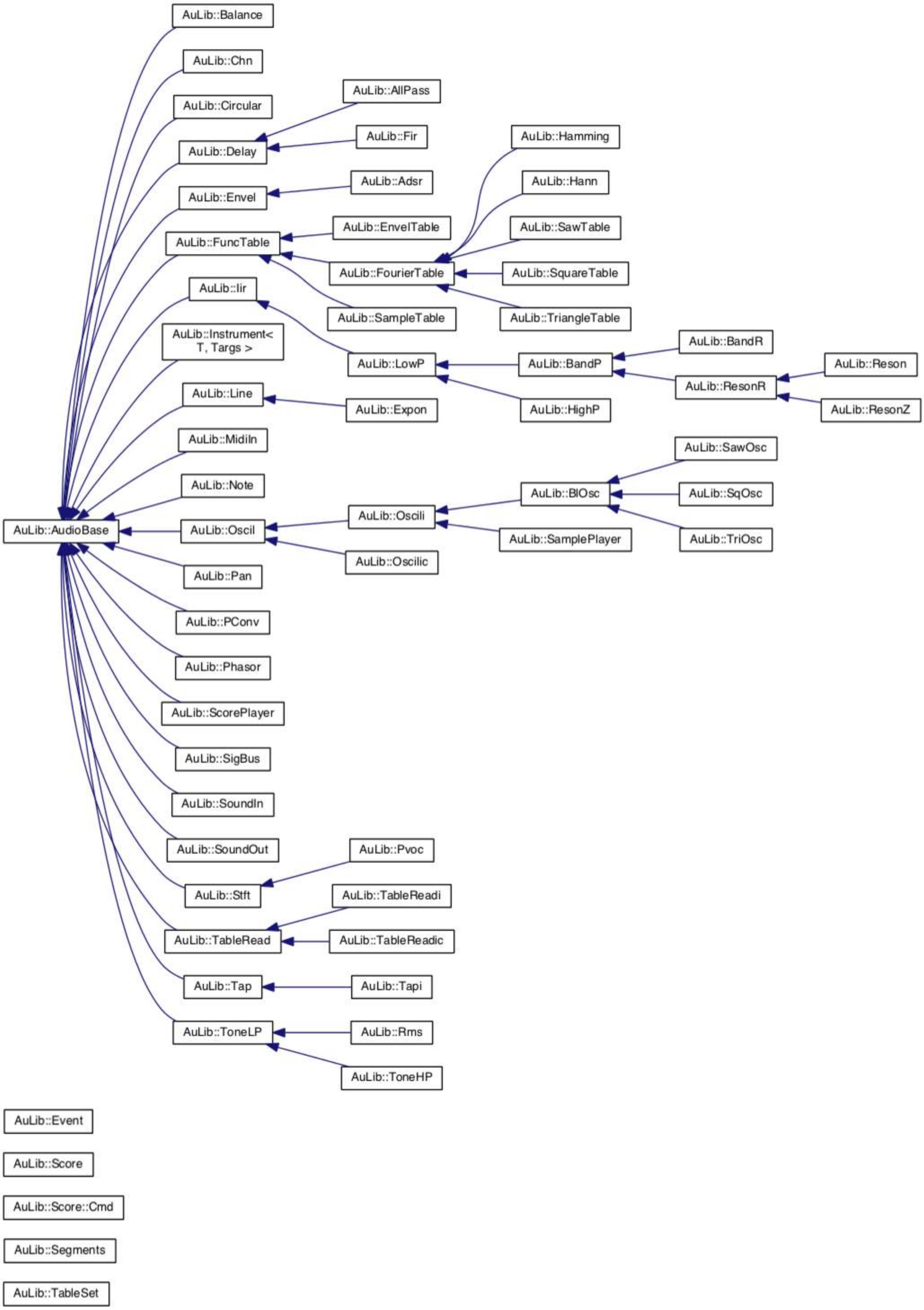


Figure 2. AuLib class hierarchy.

Equally, a processing graph based on library objects can be incorporated in a variety of settings, such as embedded hardware, mobile devices, etc..

In addition to Audio, a MidiIn class is also provided, dependent on the PortMidi library[5]. This is used to create MIDI-based applications, using the `Instrument` class.

4. EXAMPLES

In this section, we will look in detail at two examples of standard DSP algorithms and how they are implemented in the library.

4.1 Table-lookup oscillator

The table look-up oscillator, whose pedigree can be traced back to MUSIC III [9], is one of the cornerstones of digital synthesis systems. It is defined by the pair of equations 3 and 4, which characterise table lookup and phase increment, respectively, and represent a generalisation of eq.1 (N is the function size in samples and f_s is the sampling rate; $a(t)$ and $f(t)$ are the amplitude and frequency parameters).

$$s(t) = a(t)\mathrm{T}(\omega(t)) \tag{3}$$

$$\omega(t+1) = f(t)N/f_s \tag{4}$$

The function T() in eq.3 can be implemented in various forms. The simplest of them employs a floor operation, $\lfloor \omega(t) \bmod \mathrm{N} \rfloor$. This is called a truncating lookup and is implemented in listing 5, for the base class `Oscil`. Other forms can include interpolation schemes of various orders. In AuLib, linear and cubic lookups are implemented in derived classes, `Oscili` and `Oscilic` (fig.3).

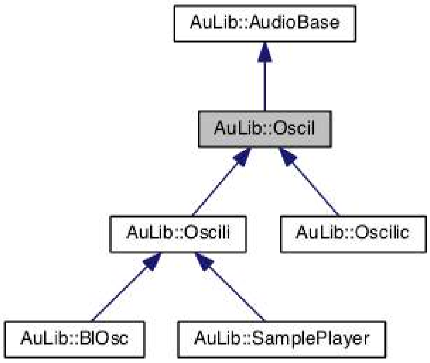


Figure 3. The `Oscil` class and its parents and children.

Listing 5. Truncating table-lookup oscillator.

```
void
AuLib::Oscil::dsp(){
 for(uint32_t i = 0;
         i < m_vframes; i++){
  am_fm(i);
    m_vector[i] =
     m_amp * m_table[(uint32_t)m_phs];
    mod();
   }
 }
```

Given that these share most components and only differ in how the table is accessed, it makes sense to express this commonality by inlining some operations. The functions $f(t)$ and $a(t)$ are updated from inputs in `am_fm()`, where they can vary on a sample-by-sample or vector-by-vector basis.

In order to give full flexibility and efficiency in parameter handling, a set of overloaded `process()` methods are provided, for fixed, varying, or modulating frequency and/or amplitude parameters. The actual processing code is delegated to `dsp()` virtual method (shown in listing 5 for the truncating case), which is then called by the `process()` interfaces defined in the base class.

4.2 Convolution

As a second implementation, we will have a look at direct (delay line) convolution, defined in eq. 5 for an impulse response N samples long. This effectively implements a finite impulse response (FIR) filter, which gives the name to the class containing the algorithm (`Fir`). Given that its main structure is a delay line, we inherit all of its internal components from `Delay` (fig.4).

$$y(t) = \sum_{n=0}^{N-1} s(t-n)h(n) \tag{5}$$

All we need to do is override the `dsp(const double*)` method. As in the previous example, and indeed across all of the library, signal processing is delegated to this function by the interface.

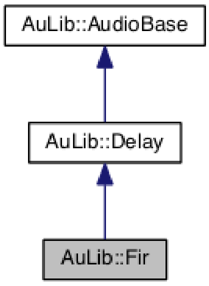


Figure 4. The `Fir` class and its parents.

The implementation is shown in listing 6, which is very compact. We can think of it as a fixed delay line tapped at each sample, with scaling factors taken from an impulse response (provided by a buffer/table object). Since the loop goes accessing the samples from maximum to no delay, the impulse response has to be read in reverse order.

Listing 6. Convolution implementation.

```
const double *
```

[5] http://portmedia.sourceforge.net/portmidi

```
AuLib::Fir::dsp(const double *sig){
 double out = 0;
 uint32_t N = m_ir.tframes();
 for(uint32_t i = 0;
        i < m_vframes; i++){
  m_delay[m_pos] = sig[i];
  m_pos = m_pos != N-1 ?
                   m_pos + 1 : 0;
   for(uint32_t j = 0,rp = m_pos;
                j < N; j++){
    out +=
     m_delay[rp] * m_ir[N-1-j];
    rp = rp != N ? rp + 1 : 0;
   }
   m_vector[i] = out;
   out = 0.;
  }
  return vector();
}
```

5. AN ECHO PROGRAM

To complete the discussion, an echo program is shown in listing 7. Fig.5 shows the flowchart for a single audio input channel (multichannel input is allowed, with stereo output).

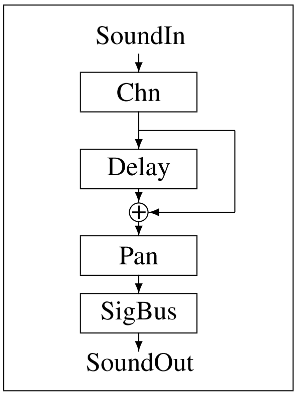


Figure 5. The signal flowchart for a single input channel in the echo program.

At the top of the program listing, lines 1 to 6 include the required AuLib classes, `SoundOut`, `SoundIn`, `Delay`, `Chn`, `SigBus` and `Pan`. A sound input object is constructed in line 17, taking as a source either the name of a soundfile, `"adc"` for realtime audio, or `"stdin"` for standard input. Arrays of channel readers, delay lines and panners are created next according to the requested number of channels. Lines 24 and 25 contain the mixer object and sound output constructors. Again, the output destination can be a soundfile name, `"dac"` for realtime, or `"stdout"` for standard output.

The processing loop, lines 33 to 44, is made up of calls to the various DSP objects through their **`operator`**`()`, plus the clearing of the signal bus mixer. It runs until the input is finished, which will depend on the source of samples. Note also the use of the overloaded operator += (in line 38) as a means of adding the dry and wet effect signals at the pan processing input.

6. CONCLUSIONS

This paper described the design of a simple, lightweight audio DSP library in C++. The main motivation is to provide a platform to develop and collect algorithms for the study, teaching and research in audio programming. The library classes are effectively thin wrappers that envelope succinct and efficient implementations of DSP operations. The code has been designed to be robust enough for general-purpose deployment in audio processing applications. Source code and multi-platform build scripts are provided at

`https://github.com/vlazzarini/aulib`

AuLib is free software, licensed by the Lesser GNU Public License.

7. REFERENCES

[1] V. Lazzarini and F. Accorsi, “Designing a sound object library,” in *Proceedings of the XVIII Brazilian Computer Society Conference*, vol. III, Belo Horizonte, 1998, pp. 95–104.

[2] P. Cook and G. Scavone, “The synthesis toolkit (stk),” in *Proceedings of the ICMC 99*, vol. III, Berlin, 1999, pp. 164–166.

[3] V. Lazzarini, “The SndObj sound object library,” *Organised Sound*, no. 5, pp. 35–49, 2000.

[4] B. Stroustrup, *The C++ Programming Language*, 2nd ed. Addison-Wesley, 1991.

[5] V. Lazzarini, “Time-domain signal processing,” in *The Audio Programming Book*, R. Boulanger and V. Lazzarini, Eds. MIT Press, 2010, pp. 463–512.

[6] ISO/IEC, “ISO international standard ISO/IEC 4882:2011, programming language C++,” 2011. [Online]. Available: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=50372

[7] V. Lazzarini, “AuLib documentation, v.1.0 beta,” 2017. [Online]. Available: http://vlazzarini.github.io/aulib/

[8] ISO/IEC, “ISO international standard ISO/IEC 14882:2014, programming language C++,” 2014. [Online]. Available: http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=64029

[9] V. Lazzarini, “The development of computer music programming systems,” *Journal of New Music Research*, no. 42, pp. 97–110, 2013.

Listing 7. Echo example program.

```
#include <Chn.h>
#include <Delay.h>
#include <Pan.h>
#include <SigBus.h>
#include <SoundIn.h>
#include <SoundOut.h>
#include <iostream>
#include <vector>

using namespace AuLib;
using namespace std;

int main(int argc, const char **argv) {

  if (argc > 2) {

    SoundIn input(argv[1]); // audio input

    std::vector<Chn> chn(input.nchnls()); // input channels
    std::vector<Delay> echo(input.nchnls(),
      Delay(0.5, 0.75, def_vframes, input.sr())); // delay lines
    std::vector<Pan> pan(input.nchnls()); // stereo panning

    SigBus mix(1. / input.nchnls(), 0., false, 2); // mixing bus
    SoundOut output(argv[2], 2, def_vframes, input.sr()); // audio output
    uint64_t end = input.dur() + 5*output.sr();

    std::vector<uint32_t> channels(input.nchnls()); // list of channels
    std::iota(channels.begin(), channels.end(), 0);

    cout << Info::version();

    while ((end -= def_vframes) > def_vframes) {
      input();
      for(uint32_t channel : channels) {
       chn[channel](input, channel + 1);
       echo[channel](chn[channel]);
       pan[channel](echo[channel] += chn[channel],
                    (1 + channel) * input.nchnls() / 2.);
       mix(pan[channel]);
      }
      output(mix);
      mix.clear();
    }

    return 0;
  } else
    std::cout << "usage: " << argv[0] << " <source> <dest>\n";

  return 1;
}
```

TOWARDS THE INTERNET OF MUSICAL THINGS

Luca Turchet
Department of Network
and Systems Engineering
KTH Royal Institute of Technology
turchet@kth.se

Carlo Fischione
Department of Network
and Systems Engineering
KTH Royal Institute of Technology
carlofi@kth.se

Mathieu Barthet
Centre for Digital Music
Queen Mary University of London
m.barthet@qmul.ac.uk

ABSTRACT

In this paper we propose to extend the concept of the Internet of Things to the musical domain leading to a subfield coined as the Internet of Musical Things (IoMUT). IoMUT refers to the network of computing devices embedded in physical objects (Musical Things) dedicated to the production and/or reception of musical content. Musical Things, such as smart musical instruments or smart devices, are connected by an infrastructure that enables multidirectional communication, both locally and remotely. The IoMUT digital ecosystem gathers interoperable devices and services that connect performers and audiences to support performer-performer and audience-performers interactions, not possible beforehand. The paper presents the main concepts of IoMUT and discusses the related implications and challenges.

1. INTRODUCTION

Recent years have seen a substantial increase in smart devices and appliances in the home, office and other environments that connect wirelessly through local networks and the Internet. This is the manifestation of the so-called *Internet of Things* (IoT), an umbrella term encompassing the augmentation of everyday physical objects using information and communication technologies. In the Internet of Things, *Things* refer to embedded systems that are connected to the Internet, which are able to interact with each other and cooperate with their neighbours to reach common goals [1]. The core technology enabling the IoT consists of wireless sensors networks (WSNs) [2]. These are networks of tiny autonomous sensor and actuator nodes that can be embedded in any physical object for control and monitoring via wireless transmission. To date, however, the application of IoT technologies in musical contexts has received little attention compared to other domains such as consumer electronics, healthcare, smart cities, and geospatial analysis [3].

In this position paper we propose to extend the concept of IoT to the musical domain leading to a subfield that we coin as the *Internet of Musical Things* [1] (IoMUT). Through its technological infrastructure, the IoMUT enables an ecosystem of interoperable devices connecting performers and audiences, to support novel performer-performer, audience-performers and audience-audience interactions.

Section 2 examines works and technologies related to the envisioned IoMUT. In Section 3, we argue for the prospect of a holistic integration of these technologies to create the envisioned IoMUT. Sections 4 and 5 discuss the implications and the current major challenges to establish the IoMUT, and Section 6 concludes the paper.

2. RELATED WORK

In this section we review key related works on which our IoMUT vision is founded.

2.1 IoT technologies

Wireless sensors networks (WSNs) design has been the object of much research, both in academia and industry (e.g., [5]). This has resulted in the definition of new communication protocols for WSNs, especially for low data rate and low power consumption, such as IEEE [6], Zigbee [2], ROLL [3]. Very recently, researchers are investigating the integration of WSNs with future wireless cellular networks, the so called 5G networks. The state-of-the art of the activity is the narrow-band IoT (NBIoT) [7].

Unfortunately, most of these protocols are not favourable for the networking of musical instruments. While the most cutting-edge networks (e.g., the fifth generation of cellular wireless networks) will deliver very high data rates, they will also provide communication delays of the order of 25ms [7]. The interconnection of musical instruments poses stringent requirements in terms of end-to-end latency to transmit and receive messages (which will have to be of the order of milliseconds [8,9]), and the reliability or probability of successful message receptions (which will have to be of the oder of 10^{-10} bit error probability). However, such requirements are not only for the application of IoT to the networked music domain, but also for emerging classes of services, such as telepresence, virtual reality, and

[1] The term “Internet of Musical Things” (or “Internet of Music Things”) [4] has previously been employed in the context of specific semantic audio applications as part of the EPSRC FAST-IMPACt project (www.semanticaudio.ac.uk), or as challenge to develop wearable instruments in hack sessions. However, to the best of our knowledge, what it entails has not been formalised in the wider context of audience/performer interactions, which is the aim of the initiative described in this work.

[2] www.zigbee.org

[3] www.ietf.org/dyn/wg/charter/roll-charter.html

mission-critical control. This is pushing the development of the emerging paradigm of "Tactile Internet" [10] and millimeter Waves (mmWaves) communications [11, 12].

Tactile Internet refers to an Internet network where the communication delay between a transmitter and a receiver would be so low that even information associated to human touch, vision, and audition could be transmitted back and forth in real-time, making remote interaction experiences, virtual or human, effective, high-quality and realistic. This novel area focuses on the problem of designing communication networks, both wireless and wired, capable to ensure ultra-low latency communications, with end-to-end delays of the oder or few milliseconds. The technological vision of Tactile Internet is that the round-trip time to send information from a source to a destination and back experiences latencies below 5ms. An important aspect of the Tactile Internet vision is the reliability and low latency for wireless communications. One of the emerging technology for the wireless communication is mmWaves. Such a technology uses wireless frequencies within the range of 10 to 300 GHz and offers data rates of giga bits per seconds over short distances. These frequencies make possible the design of small antennas that can be easily embedded in musical instruments (as proposed in the Smart Instruments concept [13]). Moreover, the high data rates will enable the transmission of multimodal content in high resolution.

2.2 Digital ecosystems

Digital ecosystems are the result of recent developments of digital network infrastructure inheriting from principles of ecological systems [14]. They are collaborative environments where species/agents form a coalition to reach specific goals. Value is created by making connections through collective ("swarm") intelligence and by promoting collaboration. The concepts underlying digital ecosystems match well the proposed goals of the Internet of Musical Things where multiple actors, smart devices and intelligent services interact to enrich musical experiences.

2.3 Networked music performance systems

Networked music performance (NMP) systems were proposed to enable collaborative music creation over a computer network and have been the object of scientific and artistic investigations [9, 15–17]. A notable example is the ReacTable [18], a tangible interface consisting of a table capable of tracking objects that are moved on its surface to control the sonic output. The ReacTable allows multiple performers to simultaneously interact with objects either placed on the same table or on several networked tables in geographically remote locations.

Networked collaborative music creations can occur over a Wide Area Network (WAN), or a Local Area Network (LAN) and in particular over a wireless one (WLAN) [17], and different methods have been proposed for each of these configurations. In [9], the authors provide a comprehensive overview of hardware and software technologies enabling NMP, including low-latency codecs, frameworks, protocols, as well as perceptually relevant aspects.

2.4 Participatory live music performance systems

Within interactive arts, participatory live music performance (PLMP) systems capitalising on information and communication technologies have emerged to actively engage audiences in the music creation process [19]. These systems disrupt the traditional unidirectional chain of musical communication from performers to audience members who are "passive" from the creative point of view (see e.g., [19–22]). Interaction techniques for technology-mediated audience participation have been proposed exploiting a wide range of media and sensors, from mobile devices [19–21, 23, 24] to tangible interfaces [25] such as light sticks [26] (see [19] for a review and classification framework).

Most PLMP systems require the audience to use a single type of device and application. Nevertheless, different types of devices could be exploited simultaneously to enrich interaction possibilities. To date, audience creative participation has mainly been based on manual controls or gestures using smartphones (e.g., screen touch, tilt). Expressive modalities could be increased by tracking physiological parameters (e.g., electrodermal activity, heart rate) [27] [28] at the individual and collective levels using devices specifically designed for this purpose, or by tracking more complex audience behaviors and body gestures. Furthermore, means of interaction in current PLMP systems typically rely on the auditory or visual modalities, while the sense of touch has scarcely been explored to create more engaging musical experiences.

2.5 Smart Instruments

Recently, a new class of musical instruments has been proposed, the *Smart Instruments* [13]. In addition to sensor and actuator enhancements provided in the so-called *augmented instruments* [29, 30], Smart Instruments are characterised by embedded computational intelligence, a sound processing and synthesis engine, bidirectional wireless connectivity, an embedded sound delivery system, and a system for feedback to the player. Smart Instruments bring together separate strands of augmented instruments, networked music and Internet of Things technology, offering direct point-to-point communication between each other and other portable sensor-enabled devices, without need for a central mediator such as a laptop. Interoperability is a key feature of Smart Instruments, which are capable of directly exchanging musically relevant information with one another and communicating with a diverse network of external devices, including wearable technology, mobile phones, virtual reality headsets and large-scale concert hall audio and lighting systems.

The company MIND Music Labs [4] has recently developed the *Sensus Smart Guitar* [13, 31] which, to the best of our knowledge, is the first musical instrument to encompass all of the above features of a Smart Instrument. Such an instrument is based on a conventional electroacoustic guitar that is augmented with IoT technologies. It involves several sensors embedded in various parts of the instrument, which allow for the tracking of a variety of

[4] www.mindmusiclabs.com

gestures of the performer. These are used to modulate the instrument's sound thanks to an embedded platform for digital audio effects. Acoustical sounds are produced by the instrument itself by means of an actuation system that transforms the instrument's resonating wooden body into a loudspeaker. Furthermore, the *Sensus Smart Guitar* is equipped with bidirectional wireless connectivity, which makes possible the transmission and reception of different types of data from the instrument to a variety of smart devices and vice versa.

2.6 Smart wearables

The last decade has witnessed a substantial increase in the prevalence of wearable sensor systems, including electronic wristbands, watches and small sensor tokens clipped to a belt or held in a pocket. Many of such devices (e.g., Fitbit[5]) target the personal health and fitness sectors. They typically include inertial measurement units (IMUs) for capturing body movement and sensors for physiological data (e.g., body temperature, galvanic skin response, heart rate). Such devices, here referred to as *smart wearables*, include wireless communication options to link to mobile phones or computers. In some cases, a small display, speaker or tactile actuator may be included. A distinguishing characteristic of wearable devices is their unobtrusiveness: they are designed to be worn during everyday activity and to passively collect data without regular intervention by the user. Such features make these devices suitable to track and collect body movements and physiological responses of audience members during live concerts. However, to date, this challenge has been scarcely addressed.

Moreover, to date, the use of the wearable devices exploiting the tactile channel in musical applications has been rather limited. Noticeable exceptions are *Rhytm'n'shoes*, a wearable shoe-based audio-tactile interface equipped with bidirectional wireless transmission [32], and *Mood Glove*, a glove designed to amplify the emotions expressed by music in film through haptic sensations [33].

2.7 Virtual reality, augmented reality, and 360° videos

The last two decades have seen an increase of both academic and industrial research in the fields of virtual reality (VR) and augmented reality (AR) for musical applications. Several virtual musical instruments have been developed (for a recent review see [34]), while musical instruments augmented with sensors, such as the Sensus Smart Guitar, have been used to interactively control virtual reality scenarios displayed on head-mounted-displays (HMD) [31].

AR has been used to enhance performance stages for augmented concert experiences, as well as for participatory performances applications. Mazzanti et al. proposed the *augmented stage* [35] an interactive space for both performers and audience members, where AR techniques are used to superimpose a performance stage with a virtual environment, populated with interactive elements. Spectators contribute to the visual and sonic outcome of the performance by manipulating virtual objects via their mobile phones. Berthaut et al. proposed *Reflets*, a mixed-reality environment that allows one to display virtual content on stage, such as 3D virtual musical interfaces or visual augmentations of instruments and performers [36]. Poupyrev et al. proposed the *augmented groove*, a musical interface for collaborative jamming where AR, 3D interfaces, as well as physical, tangible interaction are used for conducting multimedia musical performance [37].

Immersive virtual environments have been proposed as a means to provide new forms of musical interactions. For instance, Berthaut et al. proposed the *3D reactive widgets*, graphical elements that enable efficient and simultaneous control and visualisation of musical processes, along with *Piivert*, an input device developed to manipulate such widgets, and several techniques for 3D musical interaction [38,39].

The growing availability of 360° videos has recently opened new opportunities for the entertainment industry, so that musical content that can be delivered through VR devices offering experiences unprecedented in terms of immersion and presence. Recent examples include Orchestra VR, a 360° 3D performance featuring the opening of Beethoven's Fifth Symphony performed by the Los Angeles Philharmonic Orchestra, accessible via an app for various VR headsets[6] , Paul McCartney's 360 cinematic concert experience app allowing the experience of recorded concerts with 360° video and 3D audio using Google's Cardboard HMD, Los Angeles Radio Station KCRW, which launched a VR App for "intimate and immersive musical performances"[7] and FOVE's Eye Play The Piano project, which allows disabled children to play a real acoustic piano using eye tracking technologies embedded in HMDs[8] .

3. THE INTERNET OF MUSICAL THINGS

The proposed Internet of Musical Things (IoMUT) relates to the network of physical objects (*Musical Things*) dedicated to the production, interaction with or experience of musical content. Musical Things embed electronics, sensors, data forwarding and processing software, and network connectivity enabling the collection and exchange of data for musical purpose. A Musical Thing can take the form of a Smart Instrument, a Smart Wearable, or any other smart device utilised to control, generate, or track responses to music content. For instance, a Smart Wearable can track simple movements, complex gestures, as well as physiological parameters, but can also provide feedback leveraging the senses of audition, touch, and vision.

The IoMUT arises from (but is not limited to) the holistic integration of the current and future technologies mentioned in Section 2. The IoMUT is based on a technological infrastructure that supports multidirectional wireless communication between Musical Things, both locally and remotely. Within the IoMUT, different types of devices for performers and audience are exploited simultaneously to enrich interaction possibilities. This multiplies affordances and ways to track performers' and audience mem-

[5] `www.fitbit.com`

[6] `www.laphil.com/orchestravr`

[7] `www.kcrw.com/vr`

[8] `www.eyeplaythepiano.com/en`

bers' creative controls or responses. The technological infrastructure of the IoMUT consists of hardware and software (such as sensors, actuators, devices, networks, protocols, APIs, platforms, clouds, services), but differently from the IoT, these are specific to the musical case. In particular, for the most typical case of real-time applications, the infrastructure ensures communications with low latency, high reliability, high quality, and synchronization between connected devices.

Such an infrastructure enables an ecosystem of interoperable devices connecting performers with each other, as well as with audiences. Figure 1 shows a conceptual diagram of the different components that are interconnected in our vision of the IoMUT ecosystem. As it can be seen in the diagram, the interactions between the human actors (performers and audience members) are mediated by Musical Things. Such interactions can be both co-located (see blue arrows), when the human actors are in the same physical space (e.g., concert hall, public space), or remote, when they take place in different physical spaces that are connected by a network (see black arrows).

Regarding co-located interactions, these can be based on point to point communications between a Musical Thing in possession of the performer and a Musical Thing in possession of a audience member (see the blue dashed arrows), but also between one or more Musical Things of the performers towards one or more Musical Things for the audience as a whole, and vice versa (see the blue solid line arrow). An example of the latter case could be that of one or more Smart Instruments affecting the lighting system of a concert all. Regarding remote interactions, these can occur not only between audience members/performers present at the concert venue and remote audience members/performers (see the solid black arrows), but also between remote audience members/performers (see the black dashed arrows).

The communication between Musical Things is achieved through APIs (application programming interfaces, indicated in Figure 1 with the small red rectangles), which we propose could be based on a unified API specification (the IoMUT API specification). The interactions mentioned above, based on the exchange of multimodal creative content, are made possible thanks to Services (indicated with the green areas). For instance, these can be services for creative content analysis (such as multi-sensor data fusion [40], music information retrieval [41]), services for creative content mapping (between analysis and devices), or services for creative content synchronization (between devices). In particular, the implementation of novel forms of interactions that leverage different sensory modalities makes the definition of *Multimodal Mapping Strategies* necessary. These strategies consist of the process of transforming, in real-time, the sensed data into control data for perceptual feedback (haptic, auditory, visual).

4. IMPLICATIONS

Thanks to the IoMUT it is possible to reimagine the live music performance art and music teaching by providing a technological ecosystem that multiplies possibilities of interaction between audiences, performers, students, teachers, as well as their instruments and machines. This has the potential to revolutionise the way to experience, compose, and learn music, as well as even record it by adding other modalities to audio. In particular, IoMUT has the potential to make NMP and PLMP more engaging and more expressive, because it uses a radically novel approach to address the fundamental obstacles of state-of-the-art methods, which hinder efficient, meaningful and expressive interactions between performers and between performers and audiences.

The IoMUT ecosystem can support new performers-performers and audience-performers interactions, not possible beforehand. Examples of use cases include novel forms of: jamming (e.g., using apps running on smartphones to control the sound engine of a smart instrument); enhanced concert experiences (e.g., audience members in possession of haptic feedback smart wearables "feel" the vibrato of a smart violin or the rhythm of a smart drum; the emotional response of audience is used to control the timbre of Smart Instruments or the behavior of stage equipment such as projections, smoke machines, lights); remote rehearsals (point-to-point audio streaming between Smart Instruments).

To date, no human computer interaction system for musical applications enables the many interaction pathways and mapping strategies we envision in the IoMUT: one-to-one, one-to-many, many-to-one, many-to-many, in both co-located and remote situations. In contrast to traditional acoustic instruments, the IoMUT framework allows to establish "composed electronic instruments" where the control interface and the process of sound production is decoupled [42]. New situations of "performative agency" [42] can be envisioned by letting audience members and the intelligence derived within the IoMUT digital ecosystem influence the outcome of specific musical performances.

By applying the IoT to music we envision to go from the traditional musical chain (i.e., composers writing musical content for performers, who deliver it to a unique and "creatively passive" audience) to a musical mesh where possibilities of interactions are countless. We envision both common (co-located participatory music performance in a concert hall) and extreme scenarios (massive open online music performance gathering thousands or hundreds of thousands of participants in a virtual environment).

Combining such IoMUT musical mesh model with VR/AR applications it is possible to enable new forms of music learning, co-creation, and immersive and augmented concert experiences. For instance, a violin player could see through AR head-mounted-displays semantic and visual information about what another performer is playing, or be able to follow the score without having to look at a music stand. An audience member could virtually experience to walk on stage or feel in the "skin" of a performer. A whole audience could affect the lighting effects on stage based on physiological responses sensed with wireless smart wristbands. Such smart wristbands could also be used to understand audiences' affective responses. An audience could engage in the music creation process at specific times in a performance as prepared in a composer's score. A concert-

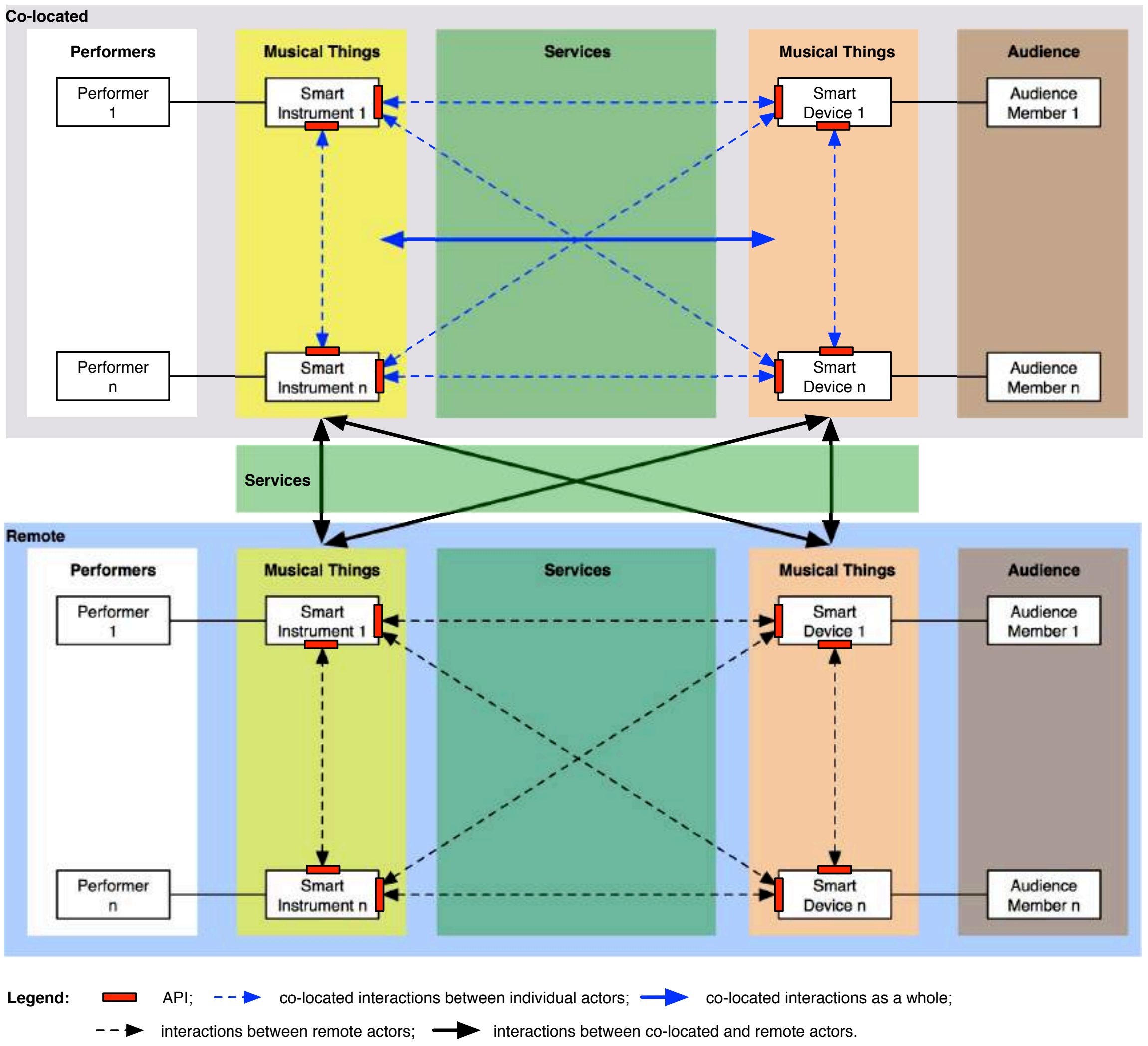


Figure 1. Block diagram of the IoMUT ecosystem.

goer could have access to “augmented programme notes” guiding and preparing them prior to the concert experience by learning more about historical and compositional aspects and listening to different renderings interactively, and letting them see the evolution in the score as the music is being played or additional information about the soloist during the concert.

In a different vein, the IoMUT has the potential to generate new business models that could exploit the information collected by Musical Things in many ways. For instance, such information could be used to understand customer behaviour, to deliver specific services, to improve products and concert experiences, and to identify and intercept the so-called “business moments” (defined by Gartner Inc.[9]).

5. CURRENT CHALLENGES

The envisioned IoMUT poses both technological and artistic or pedagogical challenges. Regarding the technological challenges, it is necessary that the connectivity features of the envisioned Musical Things go far beyond the state-of-the-art technologies available today for the music domain. Between a sensor that acquires a measurement of a specific auditory, physiological, or gestural phenomenon, and the receiver that reacts to that reading over a network, there is a chain of networking and information processing components, which must be appropriately addressed in order to enable acceptable musical interactions over the network. Nevertheless, current NMP systems suffer from transmission issues of latency, jitter, synchronization, and audio quality: these hinder real-time interactions that are essential to collaborative music creation [9]. It is also important to notice that for optimal NMP experiences, several aspects of musical interactions must be taken into account beside the efficient transmission of audio content. Indeed, during co-located musical interactions musicians rely on several modalities in addition to the sounds generated by their instruments, which include for instance the visual feedback from gestures of other performers, related tactile sensations, or the sound reverberation of the space [43]. However, providing realistic performance conditions over a network represents a significant engineering challenge due to the extremely strict requirements in terms of network latency and multimodal content quality, which are needed to achieve a high-quality interaction experience [44].

[9] www.gartner.com/newsroom/id/2602820

The most important specific technological challenges of IoMUT against the more general ones from the IoT are the requirements of very short latency, high reliability, high audio/multimodal content quality, and synchronization to be ensured for musical communication. This implies the creation of a technological infrastructure that is capable of transmitting multimodal content, and in particular audio, from one musician to another/others not only in hi-fi quality, but also with a negligible amount of latency, which enable performers to play in synchronous ways. Current IoT scientific methods and technologies do not satisfy these tight constraints needed for the real-time transmission of audio/multimodal content both at short and at large distances [9, 17]. The envisioned Tactile Internet [10] is expected to solve at least in part such issues.

The establishing of the Tactile Internet vision will require, however, the redesign of networking protocols, from the physical layer to the transport layer. For example, at the physical layer, messages will have to be inevitably short to ensure the desired low latencies, and this will pose restrictions to the data rates. At the routing layer, the protocols will have to be optimized for low delays rather than for high throughput of information. Possible avenues for future research include: the identification of all the components of a high-performance network infrastructure that is physically capable to deliver low-latency across the radio as well as in the core network segments; the proposition of new distributed routing decision methods capable to minimize the delay by design; the investigation of new dynamic control and management techniques based on optimization theory capable to configure and allocate the proper data plane resources across different domains to build low-latency end-to-end services.

The IoMUT digital ecosystems can benefit from ongoing work in the semantic web. Metadata related to multimodal creative content can be represented using vocabulary defined in ontologies with associated properties. Such ontologies enable the retrieval of linked data within the ecosystem driven by the the needs of specific creative music services (e.g., a service providing vibrato of notes played by a guitar player to enact events in the Musical Thing of an audience member). Alignment and mapping/translation techniques can be developed to enable “semantic information integration” [14] within the IoMUT ecosystem.

An important aspect of the IoMUT regards the interconnection of different types of devices. Such devices target performers or audiences (both co-located and remote), and are used to generate, track, and/or interpret multimodal musical content. This poses several other technological challenges. These include the need for ad-hoc protocols and interchange formats for musically relevant information that have to be common to the different Musical Things, as well as the definition of common APIs specifically designed for IoMUT applications.

Wearable systems present many opportunities for novel forms of musical interaction, especially involving multiple sensory modalities. Related design challenges concern the optimization for musical qualities of the sensor and actuators capabilities of these devices (e.g., temporal precision, low latency, synchronicity of audio, visual, and tactile modalities). Another related challenge is how to effectively use multiple sensory modalities in PLMP systems. In particular the haptic one leveraged by Smart Wearables could have a high impact potential on the musical experience of audience.

A major challenge to the approach of *Multimodal Mapping Strategies*, consists of how to determine mappings flexible enough to allow for musical participation expressive and meaningful to both experts and novices. These mappings could be based on features extracted in real-time from sensors data and musical audio analysis. *Multi-sensor data fusion techniques* [40] could be exploited for this purpose, which explicitly account for the diversity in acquired data, (e.g., in relation to sampling rates, dimensionality, range, and origin).

Moreover, the IoMUT demands new analytic approaches: new analytic tools and algorithms are needed to process large amounts of music-related data in order to retrieve information useful to understand and exploit user behaviours.

Regarding the non-technological challenges posed by the IoMUT, from the artistic perspective, previous attempts to integrate audiences into performances have not completely released the barriers inhibiting musical interactivity between performers and audiences. For a performance to be truly interactive, each member of the audience should be as individually empowered as the performers on stage. To achieve this, it is necessary to reimagine musical performances and invent new compositional paradigms that can catalyse, encompass and incorporate multiple and diverse contributions into a coherent and engaging whole. This poses several challenges: how can we compose music that gives individual freedom to several (potentially hundreds or thousands) of performers so that they feel empowered without compromising the quality of the performance? How do we manage all of these inputs from individuals who have different backgrounds, sensibilities and skills and leverage them so that the result is satisfying for all of them and in which each person can still recognize his or her own individual contributions? How do musicians compose and rehearse for a concert where they do not fully control the end result? In short, how do we use technology to integrate each individual expression into an evolving whole that binds people together?

A framework such as the IoMUT and what it entails for artistic and pedagogical agendas will require to be assessed. This could pave the way for novel research on audience reception, interactive arts, education and aesthetics. Such research would for instance help to reflect on the roles of constraints, agency and identities in the participatory arts. Finally, issues related to security and privacy of information should also be addressed, especially if such system was to be deployed for the masses.

6. CONCLUSIONS

IoMUT relates to wireless networks of *Musical Things* that allow for interconnection of and interaction between performers, audiences, and their smart devices. Using IoMUT we proposed a model transforming the traditional linear composer-performer-audience musical chain into a musical mesh interconnecting co-located and/or remote performers and audiences. Many opportunities are enabled by such a model that multiplies possibilities of interaction and communication between audiences, performers, their instruments and machines. On the other hand, the IoMUT poses both technological and non-technological challenges that we expect will be faced in upcoming years by both academic and industrial research.

Acknowledgments

The work of Luca Turchet is supported by MSCA Individual fellowship of European Union's Horizon 2020 research and innovation programme, under grant agreement No 749561. The work of Carlo Fischione is supported by the TNG SRA ICT project "TouCHES" - Tactile Cyberphysical Networks, funded by the Swedish Research Council. Mathieu Barthet acknowledges support from the EU H2020 Audio Commons grant (688382).

7. REFERENCES

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: a survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[2] W. Dargie and C. Poellabauer, *Fundamentals of wireless sensor networks: theory and practice*. John Wiley & Sons, 2010.

[3] E. Borgia, "The Internet of Things vision: Key features, applications and open issues," *Computer Communications*, vol. 54, pp. 1–31, 2014.

[4] A. Hazzard, S. Benford, A. Chamberlain, C. Greenhalgh, and H. Kwon, "Musical intersections across the digital and physical," in *Digital Music Research Network Abstracts (DMRN+9)*, 2014.

[5] A. Willig, "Recent and emerging topics in wireless industrial communication," *IEEE Transactions on Industrial Informatics*, vol. 4, no. 2, pp. 102–124, 2008.

[6] *IEEE Std 802.15.4-2996, September, Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)*, IEEE, 2006.

[7] S. Landström, J. Bergström, E. Westerberg, and D. Hammarwall, "Nb-iot: a sustainable technology for connecting billions of devices," *Ericsson Technology Review*, vol. 93, no. 3, pp. 1–12, 2016.

[8] C. Rottondi, M. Buccoli, M. Zanoni, D. Garao, G. Verticale, and A. Sarti, "Feature-based analysis of the effects of packet delay on networked musical interactions," *Journal of the Audio Engineering Society*, vol. 63, no. 11, pp. 864–875, 2015.

[9] C. Rottondi, C. Chafe, C. Allocchio, and A. Sarti, "An overview on networked music performance technologies," *IEEE Access*, 2016.

[10] G. P. Fettweis, "The Tactile Internet: applications and challenges," *IEEE Vehicular Technology Magazine*, vol. 9, no. 1, pp. 64–70, 2014.

[11] H. Shokri-Ghadikolaei, C. Fischione, G. Fodor, P. Popovski, and M. Zorzi, "Millimeter wave cellular networks: A MAC layer perspective," *IEEE Transactions on Communications*, vol. 63, no. 10, pp. 3437–3458, 2015.

[12] H. Shokri-Ghadikolaei, C. Fischione, P. Popovski, and M. Zorzi, "Design aspects of short-range millimeter-wave networks: A MAC layer perspective," *IEEE Network*, vol. 30, no. 3, pp. 88–96, 2016.

[13] L. Turchet, A. McPherson, and C. Fischione, "Smart Instruments: Towards an Ecosystem of Interoperable Devices Connecting Performers and Audiences," in *Proceedings of the Sound and Music Computing Conference*, 2016, pp. 498–505.

[14] H. Boley and E. Chang, "Digital ecosystems: Principles and semantics," in *Inaugural IEEE International Conference on Digital Ecosystems and Technologies*, 2007.

[15] Á. Barbosa, "Displaced soundscapes: A survey of network systems for music and sonic art creation," *Leonardo Music Journal*, vol. 13, pp. 53–59, 2003.

[16] G. Weinberg, "Interconnected musical networks: Toward a theoretical framework," *Computer Music Journal*, vol. 29, no. 2, pp. 23–39, 2005.

[17] L. Gabrielli and S. Squartini, *Wireless Networked Music Performance*. Springer, 2016.

[18] S. Jordà, G. Geiger, M. Alonso, and M. Kaltenbrunner, "The reactable: exploring the synergy between live music performance and tabletop tangible interfaces," in *Proceedings of the 1st international conference on Tangible and embedded interaction*, 2007, pp. 139–146.

[19] Y. Wu, L. Zhang, N. Bryan-Kinns, and M. Barthet, "Open symphony: Creative participation for audiences of live music performances," *IEEE Multimedia Magazine*, no. 48-62, 2017.

[20] G. Fazekas, M. Barthet, and M. B. Sandler, *Novel Methods in Facilitating Audience and Performer Interaction Using the Mood Conductor Framework*, ser. Lecture Notes in Computer Science. Springer-Verlag, 2013, vol. 8905, pp. 122–147.

[21] L. Zhang, Y. Wu, and M. Barthet, "A web application for audience participation in live music performance: The open symphony use case," in *Proceedings of the international conference on New Interfaces for Musical Expression*, 2016, pp. 170–175.

[22] A. Clément, F. Ribeiro, R. Rodrigues, and R. Penha, "Bridging the gap between performers and the audience using networked smartphones: the a.bel system," in *Proceedings of International Conference on Live Interfaces*, 2016.

[23] A. Tanaka, "Mobile music making," in *Proceedings of the international conference on New Interfaces for Musical Expression*, 2004, pp. 154–156.

[24] N. Weitzner, J. Freeman, S. Garrett, and Y.-L. Chen, "massmobile-an audience participation framework." in *Proceedings of the international conference on New Interfaces for Musical Expression*, 2012, pp. 21–23.

[25] B. Bengler and N. Bryan-Kinns, "Designing collaborative musical experiences for broad audiences," in *Proceedings of the 9th ACM Conference on Creativity & Cognition*. ACM, 2013, pp. 234–242.

[26] J. Freeman, "Large audience participation, technology, and orchestral performance," in *Proceedings of the International Computer Music Conference*, 2005.

[27] D. J. Baker and D. Müllensiefen, "Hearing wagner: Physiological responses to richard wagner's der ring des nibelungen," in *Proc. Int. Conference on Music Perception and Cognition*, 2014.

[28] A. Tanaka, "Musical performance practice on sensor-based instruments," *Trends in Gestural Control of Music*, vol. 13, pp. 389–405, 2000.

[29] T. Machover and J. Chung, "Hyperinstruments: Musically intelligent and interactive performance and creativity systems," in *Proceedings of the International Computer Music Conference*, 1989.

[30] E. Miranda and M. Wanderley, *New digital musical instruments: control and interaction beyond the keyboard*. AR Editions, Inc., 2006, vol. 21.

[31] L. Turchet, M. Benincaso, and C. Fischione, "Examples of use cases with smart instruments," in *Proceedings of AudioMostly Conference*, 2017 (In press).

[32] S. Papetti, M. Civolani, and F. Fontana, "Rhythm'n'shoes: a wearable foot tapping interface with audio-tactile feedback," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, 2011, pp. 473–476.

[33] A. Mazzoni and N. Bryan-Kinns, "Mood glove: A haptic wearable prototype system to enhance mood music in film," *Entertainment Computing*, vol. 17, pp. 9–17, 2016.

[34] S. Serafin, C. Erkut, J. Kojs, N. Nilsson, and R. Nordahl, "Virtual reality musical instruments: State of the art, design principles, and future directions," *Computer Music Journal*, vol. 40, no. 3, pp. 22–40, 2016.

[35] D. Mazzanti, V. Zappi, D. Caldwell, and A. Brogni, "Augmented stage for participatory performances." in *Proceedings of the international conference on New Interfaces for Musical Expression*, 2014, pp. 29–34.

[36] F. Berthaut, D. Plasencia, M. Hachet, and S. Subramanian, "Reflets: Combining and revealing spaces for musical performances," in *Proceedings of the international conference on New Interfaces for Musical Expression*, 2015.

[37] I. Poupyrev, R. Berry, J. Kurumisawa, K. Nakao, M. Billinghurst, C. Airola, H. Kato, T. Yonezawa, and L. Baldwin, "Augmented groove: Collaborative jamming in augmented reality," in *ACM SIGGRAPH 2000 Conference Abstracts and Applications*, 2000, p. 77.

[38] D. Berthaut, M. Desainte-Catherine, and M. Hachet, "Interacting with 3d reactive widgets for musical performance," *Journal of New Music Research*, vol. 40, no. 3, pp. 253–263, 2011.

[39] F. Berthaut and M. Hachet, "Spatial interfaces and interactive 3d environments for immersive musical performances," *IEEE Computer Graphics and Applications*, vol. 36, no. 5, pp. 82–87, 2016.

[40] D. Hall and J. Llinas, "Multisensor data fusion," in *Multisensor Data Fusion*. CRC press, 2001.

[41] J. Burgoyne, I. Fujinaga, and J. Downie, "Music information retrieval," *A New Companion to Digital Humanities*, pp. 213–228, 2016.

[42] O. Bown, A. Eldridge, and J. McCormack, "Understanding interaction in contemporary digital music: from instruments to behavioural objects," *Organised Sound*, vol. 14, no. 2, pp. 188–196, 2009.

[43] W. Woszczyk, J. Cooperstock, J. Roston, and W. Martens, "Shake, rattle, and roll: Getting immersed in multisensory, interactive music via broadband networks," *Journal of the Audio Engineering Society*, vol. 53, no. 4, pp. 336–344, 2005.

[44] M. Slater, "Grand challenges in virtual environments," *Frontiers in Robotics and AI*, vol. 1, p. 3, 2014.

Environment-Mediated Coupling of Autonomous Sound-Generating Systems in Live Performance: An Overview of the *Machine Milieu* Project.

Dario Sanfilippo
Reid School of Music - The University of Edinburgh
Dario.Sanfilippo@ed.ac.uk

Agostino Di Scipio
Conservatorio di L'Aquila / Université Paris-VIII
discipio@tin.it

ABSTRACT

Today, a large variety of technical configurations are used in live performance contexts. In most of them, computers and other devices act usually as powerful yet subordinated agencies, typically piloted by performers: with few notable exceptions, large-scale gestures and structural developments are left either to the performer's actions or to well-planned automations and/or composing algorithms. At the same time, the performance environment is either ignored or 'tuned out', ideally kept neutral with regard to the actual sound events and the overall performance process. This paper describes a different approach. The authors investigate the complex dynamics arising in live performance when multiple autonomous sound systems are coupled through the acoustic environment. In order to allow for more autonomous and adaptive – or better: ecosystemic *– behaviour on the part of the machines, the authors suggest that the notion of* interaction *should be replaced with that of a permanent and continuing* structural coupling *between machine(s), performer(s) and environment(s). More particularly, the paper deals with a specific configuration of two (or more) separate computer-based audio systems co-evolving in their autonomic processes based on permanent mutual exchanges through and with the local environment, i.e., in the medium of sound only. An attempt is made at defining a self-regulating, situated, and hybrid dynamical system having its own agency and manifesting its potential behaviour in the performance process. Human agents (performers) can eventually intrude and explore affordances and constraints specific to the performance ecosystem, possibly biasing or altering its emergent behaviours. In so doing, what human agents actually achieve is to specify their role and function in the context of a larger, distributed kind of agency created by the whole set of highly interdependent components active in sound. That may suggest new solutions in the area of improvised or structured performance and in the sound arts in general.*

Overall, the approach taken here allows for an empirical investigation of the mobile and porous boundaries between the kind of environmental agency connoting densely connected networks, on one hand, and the freedom (however restricted) in independent and intentional behaviour on the part of human agents intruding the ecosystem, on the other.

1. INTRODUCTION

The research documented here stems from the collaboration between two sound artists implementing and exploring complex dynamical systems in the context of live performance [1, 2, 3]. Born out of hands-on experimentation in the authors' private workspace, the collaboration started in 2014 and developed through different stages until 2016, when was given the project title *Machine Milieu*.

In usual "performance ecosystems" [4], computers and other devices act as powerful yet subordinated agencies, typically piloted by practitioners, such that large-scale gestures and structural developments arise from either performer's actions or well-planned automations and/or composing algorithms. To a large extent, the environment is either ignored or 'tuned out', perceived as ideally neutral with regard to the actual sound events and musical gestures. Our research aims instead at establishing a structural, multi-directional connection between the three main agential nodes in such a performance ecosystem performer(s), machine(s) and environment(s) as something entirely mediated by the sounding environment itself. The basic idea is to let every agency develop and sonically manifest itself as a function of all of the other agencies involved. Essential is an effort aiming at having the machines (computers) somehow make sense of what happens sound-wise in the local, shared environment, and act accordingly.

One may say, with an altogether different terminology, that the goal here is to create and situate a performative agency providing the conditions for the emergence [5] of a *self* through the perturbation and the exploration of the surrounding sound environment, the latter representing a complex structured ensemble standing for *non-selves* i.e., standing for *other selves* [6]: a coherent system develops a sense of its self as each of its components affects all of the others, while the environment to which it is structurally coupled is actively and inevitably influencing the whole process, i.e., the behaviour of the single components as well as their interactions [7]. In essence, this is a recursive process consistent with Gregory Bateson's definition of *information* as something built and processed by a (cognitive) system coupled to an environment [8]. The performative agency we aim to design and explore is a 'minimally cognitive' system [9, 10] that construes information about its surrounding in order to establish a positive, indeed constructive relationship with the forces present and active in the surroundings. (To what extent that may be pursued through 'feature-extraction' methods and 'sound descriptors' is, in our view, a central issue of theoretical as well as technical relevance, yet we will have to leave it for a more specific discussion, in another paper). In a batesonian perspective, a bit of information is notoriously defined as a ”difference which makes a difference”: a differential quantum that travels and spreads across a circuit and undergoes an overall process or recursive interactions and transformations. As an ensemble, the recursive process has no single site of agency, no specific site or centre of global and unilateral control over the ensemble or any of the single parts. However, while the interacting ’parts’ let emerge a ’whole’ a system that cannot be reduced to the single separate parts the emergent whole in turn may eventually bias or bend the individual parts in their further doing (*downward causation*) [11] and thus reinforce the ensemble in its consistent and distinct dynamical behaviour.

We call *music* the traces of such a process in sound.

It should now be clear why we are not referring to our collaboration simply as a *duo* (two human agents, each ’playing’ its own device, overlapping its own sound output to that of the partner): it would be more appropriate to think of the overall performance ecosystem explored here as an integrated, hybrid organism, an ’assemblage’ made of humans, machines and environment(s) as well as of a rich set of mediations and negotiations allowing for their interdependency. The structure of this assemblage is essentially that of a complex dynamical system [11, 12]. We believe that the realisation of a distributed and parallel interaction like the one described here, based on a tight sound relationship to the actual performance site, results in a peculiar performative approach and opens to innovative aesthetic developments in the practice of live sound art.

2. THE IMPORTANCE OF AUTONOMY AND FEEDBACK IN MUSIC SYSTEMS

The condition of mutual influence is fundamental and implicit in any notion of interaction. In our view, however, in order to achieve such a condition both humans and machines should be capable of autonomous dynamical behaviours so that they can act (sound-wise) in the environment while also adapting to the (sounding) actions of other forces and systems. In that sense, automated not to be confused with autonomous systems are not at all adequate: in designs based on pre-determined or stochastic scheduling of sonic events, for example exploiting pseudo-random number generators and other abstract formal rules, musical articulations are operated in a domain that is both independent of, and fundamentally (in)different to, the domain where musical action takes place namely, *sound*. Across the many levels of the performance ecosystem, we avoid relinquishing action to any such ’independent’ agency, and try to lean as much as possible on sites of agency situated in the experiential milieu of sound, namely as constantly mediated by the surrounding environment. On a general level, defining autonomy in music systems is a rather difficult task [13]. In a bio-cybernetic view [7, 14] autonomous agency in a domain requires a structural openness, a constant exposure to an ’external’ space. Different from automation, a fruitful notion of *autonomy* includes an operational openness to heterogeneous forces and actors in the environment: the system’s necessary closure (defining its identity, its self) consists in a loop onto itself through the environment, which provides a pool of possible sources of information, and the space where other agencies are themselves active in their autonomic process.

Leaning on independent sources of information and agency leads to a lack of contextuality and coherence between sonic contents and musical developments. An approach based on autonomous dynamical systems can reveal crucial and perhaps necessary in order to avoid a certain lack of organicity and shape up a performative situation more consistent with the metaphor of a live (living) practice and a live (lived) experience of sound and music.

The implementation of multiple feedback delay networks is a very central factor in our practice, for the peculiar dynamical characteristics they exhibit seem well suited to human-machine interactions in the context of music performance [15]. In the particular case of *Machine Milieu*, we have a distributed and structural interaction between all of the variables and domains involved in the electroacoustic chain (as one can see by following the oriented arrows in Figure 1): from microphones and loudspeakers to the digital processes, as well as to the performers and the environment itself. The recursive interactions born in the feedback loops allow the overall process to develop from the micro-time scale properties (signal contours and related timbral percepts) to the formal unfolding and behavioural transitions in the systems. Because of the nonlinearities included at various levels (due to the particular audio transformations, as well as to the circuitry of the analog transducers involved), any slight perturbation recirculates in the process in a way that can potentially have significant short and long-term effects for all of the variables and domains, resulting in a very delicate and non-trivial coexistence and binding of all components in the live environment. When detected ”differences” in the medium are truly informative (to use batesonian terminology, when information is indeed construed in the coupling of two systems), a larger process is started in the context of which sound is revealed capable of shaping itself in an organic and expressive way: thus, the agency of the overall system allows for not only the emergence of specific timbral or gestural shales, but also of larger-scale articulations (musical form).

Indeed, besides being central to any approach of physical modelling of sound [16, 17, 18], in a larger cybernetic perspective, feedback mechanisms [19] are reputed the key for the modelling of artificial forms of intelligent behaviour, perhaps aimed less at the simulation of something already existing, and more at the manifestation of emergent entities with their own personality (or, in our case, musicality). Information and computation, as referred to any entity having cognitive functions, were historically defined by Heinz von Foerster as recursive processes [20] in a system having sufficient complexity in dealing with the environment and the minimal requisite variance (as defined by W. R. Ashby [21]) in order to be stable and support its self.

3. OVERVIEW DESCRIPTION OF *MACHINE MILIEU*

3.1 Setup

The main components and relationships in the *Machine Milieu* project are schematised in Figure 1. The overall

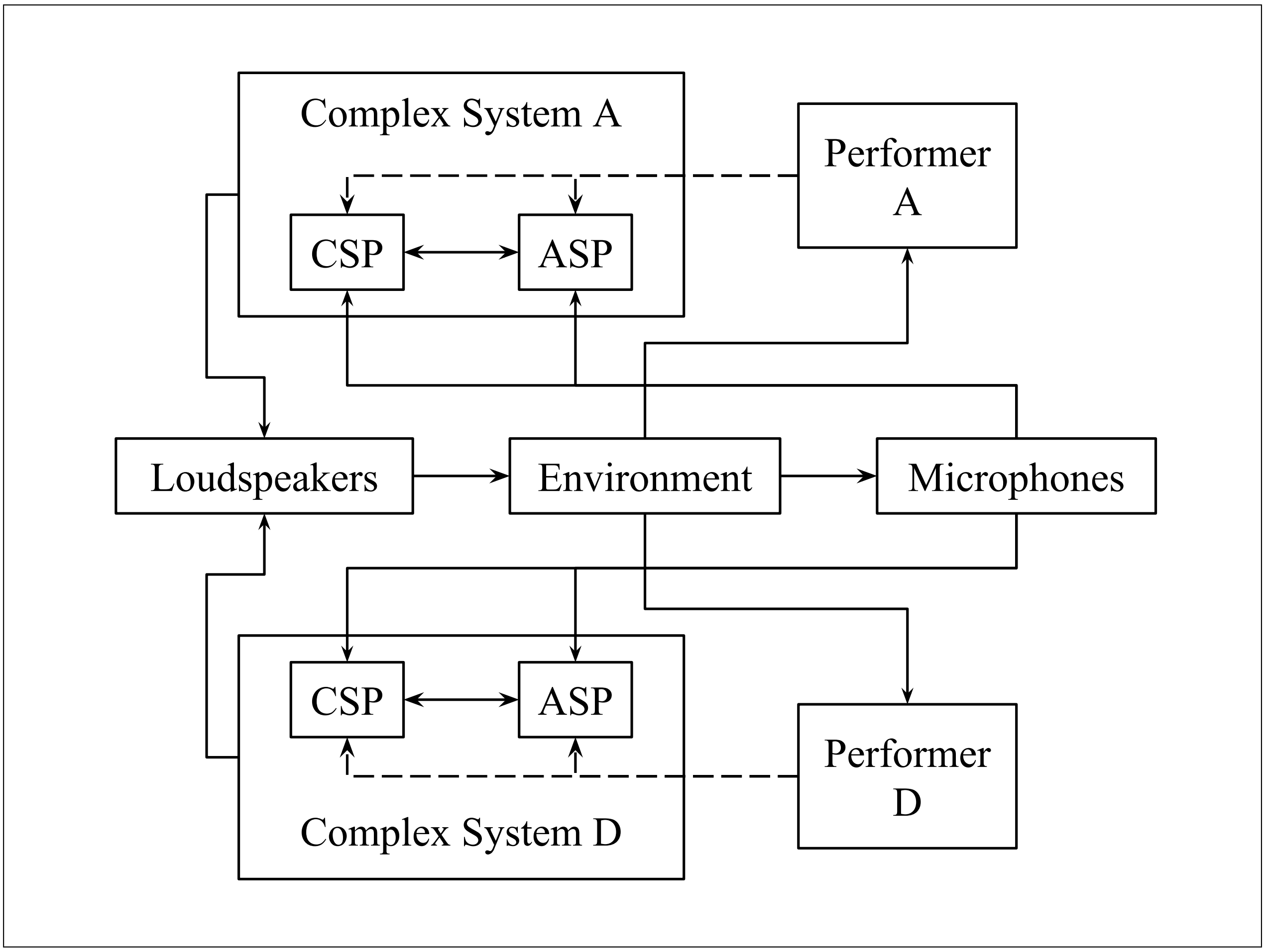


Figure 1. Overall configuration of the *Machine Milieu* project. Oriented arrows represent the signal flow across the network. By ”environment” we mean here the performance room, with its specific acoustics, i.e., the spatial niche which actually shapes the sound produced by the loudspeakers (as well as by the performers and the audience). The ”complex audio systems” are sound-generating systems consisting of time-variant networks of nonlinear processing units, interrelated through positive and negative feedback loops. Each complex system includes audio signal processing (ASP) units as well as control signal processing (CSP) units: the first include DSP algorithms meant to process sound signals; the second include DSP algorithms meant to analyse incoming signals and turn that signal descriptor data into control signals useful to drive the states variables of the ASP. These two subsystems are mutually influencing each other. By dropping the two ”performers”, we can still rely on an autonomous, unsupervised dynamical system.

setup includes two performers (called A and D); two sound-generating systems (real-time computer-operated signal processing algorithms) both including audio signal processing (ASP) as well as control signal processing (CSP) units; a set of microphones; a set of loudspeakers; and of course a performance space (Environment).

The interrelatedness among the involved components can be clarified as follows. Each of the two performers' actions is a function of the sonic context as perceived in the environment. The state of each of the two audio systems is dependent on the sonic context (captured through microphones and internally analysed) as well as on the performers' direct access to the internal values of both the audio and the control signal processing algorithms included. Finally, by making audible the computer processing output at specific positions in the local environment, the loudspeakers act as the very means that elicit the performance space's acoustical response, thus affecting both the performers and the computer processes.

In this general setup, performers may eventually be dropped out. In that case, we have an entirely autonomous and unsupervised ecosystem consisting in the coupling of computational devices, a number of transducers and the room acoustics.

3.2 Technical aspects

The complex audio systems implemented are time-variant feedback delay networks through which a number of nonlinear components are interrelated. Both positive and negative feedback mechanisms are established in order to maximise counterbalancing structures in the network and achieve higher degrees of variety in the resulting behaviours. The two audio systems could be divided into at least two subsystems, although they should always be considered as a single unit given their structural synergy. All signal processing is done through Pure Data Vanilla [1] and Kyma/Pacarana [2] , and use exclusively time-domain processing methods, with the microphone signals as inputs. (Frequency-domain methods are set aside mainly for reasons of economy in computational load.)

The ASP units include:

- asynchronous granulation

[1] http://msp.ucsd.edu/software.html

[2] http://kyma.symbolicsound.com/

- various (re)sampling methods
- waveshaping (nonlinear distortion)
- feedback and cascaded FM
- PWM pulse-width modulation
- simple high-pass and low-pass filtering, as well as all-pass and comb filtering
- delay networks (FDN, feedback delay networks)

We dispense ourselves with the details: the listed audio signal processing methods, and the related control variables, will be certainly well-known to the reader. However, it should be noted that, while normally working in the sub-audio frequency range, control variables (CSP outputs) are actually worked out here as *audio* signals in our implementation. Therefore, while maintaining their *control* function, here they are occasionally mapped in the audible range and accordingly used as modulation signals (i.e., with audible spectral changes). It might also be worth noting that FDN configurations may act in several different ways: here, they are often used to dispatch signals across the set of audio processing methods, sometimes creating recursive paths, and thus contributing to articulate layers of sonic transformations through larger time frames; yet in certain circumstances they will also act in a more typical way, i.e., as reverb units (in which case included variables are those you would expect from a reverberator unit).

The CSP units involve either simple mappings or more elaborate transformations of data created by the real-time analysis and evaluation of a limited range of sonorous aspects in the audio signal. There is today a very large body of work on 'feature-extraction' and 'descriptors' (see related topics and broader questions in [22, 23, 24]) which represents for us a relevant shared common knowledge. However, in our project we do not rely on any library of descriptors or other existing resources of the kind. Rather, we prefer implementing ourselves simple but efficient methods matching specific requirements, that of implying very limited computational load (all processes must be highly efficient under real-time computation constraints) and that of coming up with a small but varied set of hypothesis on broad but auditorily important aspects of the sound in the performance space.

In our work, the main sonic features subject to tracking algorithms include:

- loudness
- density
- brightness
- noisiness
- roughness

Loudness estimations are obtained as RMS values calculated over subsequent windowed segments of the incoming signal (as an alternative, we sometimes integrate amplitude values over the signal chunks). Windowed segments can be of different sizes, and the size can also change during the performance as a function of some other control signal. *Brightness* and *noisiness* are calculated via original CPU-efficient algorithms operating in the time domain (zero-crossing rate, differentiation, spectral median) and/or via averaged responses of large-width band-pass filters. *Roughness* estimation uses peak envelope for the analysis of transients. In contrast with those, *density* is probably a rather unusual kind of descriptor, and is understood here in terms of RMS values calculated over multiple extended signal segments (in the order of few to several seconds), eventually correlated with peak envelope tracking (attack transients).

In actuality, as anticipated above, the analysis data are for us a source out of which, with a little signal processing, we can shape up multiple *control signals*. Indeed, the observed sonic characteristics are then combined or used individually and eventually mapped (linearly or non-linearly) over value ranges compatible with those of control variables in the ASP algorithms. But they are, too, more extensively processed (via simple filters, delay units, etc.) to create a variety of viable control signals. We acknowledge a creative role in the shaping of several control signals out of one or anyway few sound sources: in this perspective indeed a broader conceptual shift is taking place from "interactive composing" to "composing the interactions" [1]. We also explore the idea of higher-order analytical data created by statistics and measurements of (lower-level) extracted information. Control signals based on higher-order data are especially useful to affect longer-term developments in the variable space, thus achieving a sense of global orientation across prolonged sound textures and global musical shapes.

Once applied to audio processing variables, control signals will affect either subtle or overt changes in the sound, and thus – because sound is the very source of control signals – they will loop back onto themselves, indirectly affecting their own subsequent unfolding. This is feedback in the low-frequency domain (second or higher-order emergent patterns heard as longer-term patterns or events). The specific final mapping is what will determine whether the relationship between variables and control signals makes for a positive or negative feedback loop.

It is important to notice that, as 'heard' by any of the individual ecosystem components, the sonic environment comprises 'one's own' sound (the particular component's) as well as the sound output of other sources in the environment (including the environment itself, if not acoustically dry or idle). Accordingly, of special interest is the generation of control signals based on feature-extraction methods somehow able to distinguish and track down, from within the total sound, those events or properties coming from one's own contribution in the total system and those coming from others.

3.3 Modes of performance

The complete resultant infrastructure can be seen as a densely interconnected network. Densely connected network systems have been fruitfully investigated in the context of algorithmically based live performance practices e.g., The Hub [25, 26] and the early League of Automatic Music Composers [27], and resurface today in collective live coding practices. In [26] a discussion is proposed as to the ecosystemic (or simply technical) nature of such networked performance situations. Here, we consider that in those examples the network interconnectedness typically relies on abstract, formal protocols of music data and their transfer along lines of digital connections (MIDI, OSC, etc.), if not on more general computer network protocols, unrelated to music. Leaning on previous works [1, 2, 3], the main sites of agency in the network are better understood as components in a *sonic* ecosystem, i.e., they are structurally coupled in the medium of sound. The individual functions as well as the interdependencies among the ecosystem components remain under the spell of the permanent mechanical (acoustical) mediation of the local environment. Taking up anthropologist Tim Ingold's critique of the widespread notion of *network* (see [28], p. 70), in our case the connections out of which a larger dynamical ensemble is formed, are not between *nodes* but rather along *lines* (acoustical propagation in the air). Understood as ensembles relinquished in a physical (and hypertechnologised) environment, and acted upon (or let unsupervised) by human agents, such situated networks of sonic interactions make it difficult to tell what is the place and meaning of 'computing' in such configurations of humans and nonhumans [29]. At the same time, they allow us to emphasise the very materiality and situatedness of algorithmic structures, quite usually considered as purely formal constructions disconnected from and independent of any sensible context [30].

By entering the otherwise unsupervised autonomous system, performers have two fundamental ways through which they can interact with the machines. On one hand, they can operate directly on the ASP and CSP, namely by manually varying the variables in the audio transformation algorithms (via a computer GUI or an external controller), or by reconfiguring the mapping and the dispatching of control signals. Alternatively, performers can for example change the position of the microphones in the performance space, or creatively modify the frequency response of microphones with small resonators such as pipes or boxes.

Either ways, relying on an attitude of improvisation certainly represents a straightforward and consistent approach when it comes to performing with feedback systems [31, 32], especially in consideration that improvisation is itself intrinsically a feedback process (current actions are mostly determined by carefully listening and promptly reacting to whatever results from earlier actions). Indeed, we have often adopted a largely improvisational approach during the *Machine Milieu* sessions we were able to have so far. However we thought that some alternative ways of going should be considered: even when 'radical', improvisation may lead to higher-level patterns and gestural behaviours which all too directly connote (and delimit) the range of possible interesting situations. More particularly, we felt that improvisation would be good in order to explore specific aspects of the network dynamics in our performance ecosystem, but could prevent us from understanding and creatively investigating other aspects, specifically with regard to the potential autonomic behaviours it may engender.

We explored two main alternatives. The first is actually just a small shift away from radical improvisation: because the two autonomous sound-generating systems were capable of exhibiting peculiar dynamical behaviours, we tried to contrast that attitude with our interventions in an attempt at forcing them to a somewhat more static behaviours (prolonged sonic textures, rich in short-term variations, but consistent and 'static' on the longer run). In systemic terms, this way of acting in the system is a form of negative feedback, and very difficult to realise: the artefacts and traces of such an almost impossible task actually resonate in the musical unfolding of the performance itself.

In the third approach, instead, there is a sharper separation between performer actions and the overall ecosystem process. The goal, in this case, is to do as least as possible in order to let the systems start sounding in the room thus eventually interacting between themselves. In this situation, each one system is actually interfering with its own sound as heard in the surrounding environment but through the other system. It was interesting to hear them creating sonic materials and gestures somewhat different form those generated when operating each for itself. After setting up the 'initial conditions' (tuning of variables, placement of microphones and speakers), we would let the overall process develop until satisfying behaviours emerged; eventually, when the potential seemed to have exhausted, we intervened slightly altering the setup and defining new initial conditions, letting the process manifest new emergent behaviours (we would then keep proceeding like that for a few times).

In retrospect, the three approaches taken are all consistent with the *Machine Milieu* project. They represent for us different ways to investigate the boundaries between, on one hand, the kind of environmental agency [14] connoting our performance ecosystem and, on the other,the (small but significant) margin of manoeuvre in the independent, targeted and intentional actions taken on by human agents. What remains constant, and central, across these different performative approaches, is the notion that the main system components involved (either including or not including humans) are permanently coupled to each other. Properly speaking, they do not interact there is no discrete event in space that can be called an interaction and that might be entirely independent in time from (few or several) earlier exchanges. Rather, there is a continuing flow of mutual exchanges or reciprocal determinations. Within an ecosystemic perspective, the notion of interaction is not at all satisfactory, and should be replaced by a notion of structural coupling (itself a term coming from general system theory, and more specifically connoting the way living systems use to deal with their environment [7]).

4. CONCLUSIVE REMARKS

In this paper, we have overviewed some artistic research issues central in the *Machine Milieu* project, in the context of which we explore the environment-mediated coupling of autonomous generative audio systems in live performance. By creating a hybrid (digital, analog and mechanical) assemblage for creatively experimenting with the interdependencies among situated autonomous systems, we try to open a space for performers and listeners to ponder questions of context awareness, ecosystemic dynamics, materiality of algorithms in daily life, while also empirically touching on more fundamental questions of autonomy and dynamical behaviour, on the other. We are inclined to consider such notions crucial in music systems intended for live performance, and this is especially true when as we do here one replaces a notion of interaction (nowadays too charged with misunderstandings and often treated in trivial ways) with the systemic notion of "structural coupling" (among machines, performers and environments). The implicit statement is that, in line with larger theoretical efforts today emphasising the environmentalisation of agency (or the agentivity of overly technologised environments) [14, 33], the realisation of a kind of distributed musical agency may result in an understanding of musical creativity as a phenomenon emergent and indeed ecosystemic.

Furthermore, the dialectical interweaving of individual action and autonomous ecosystemic processes, allows for a variety of performative practices to be explored which, on the sole basis of sound-mediated interactions, projects the interdependencies and codependencies of part and whole i.e., of systems and subsystems, as well as of structurally coupled but different domains to the level of larger scale developments in a performance: any notion of *form* becomes then closer to the ensemble of component parts either contrasted or harmoniously interwoven among them in their temporal and spatial activity. In the particular experience reported above, we have touched on this issue when contrasting the autonomic behaviours of unsupervised environment-mediated machines, against a direct and radical improvisation approach of practitioners entering the performance ecosystem.

In further work, we expect that approaches such as the one taken here may contribute to a deeper understanding of what can be considered *live* in sound art and music performance with live electronics. Different from other authors who have tackled this issue [34, 35], we are convinced that, in and of itself, the sheer presence of human performers in the context of and in the foreground of an overly technologised playground is insufficient to clarify hybrid assemblages merging human and machine agency such as deployed by in the *Machine Milieu* project. The implicit and unquestioned opposition of living (hence cognitive) vs non-living (hence non-cognitive) systems reflecting in other similar dualities, such as human vs. inhuman, natural vs. artificial, etc. is today probably untenable as a basis to understand what (or who) does live in live electronics. Maybe we should look for what is no more human in living systems, and what is already all too human in artificial (i.e., humanly construed) systems.

5. REFERENCES

[1] A. Di Scipio, "Sound is the interface: from interactive to ecosystemic signal processing," *Organised Sound*, vol. 8, no. 03, pp. 269–277, 2003.

[2] ——, "Émergence du son, son d'emergence: Essai d'épistémologie expérimentale par un compositeur," *Intellectica*, vol. 48, no. 49, pp. 221–249, 2008.

[3] D. Sanfilippo, "Turning perturbation into emergent sound, and sound into perturbation," *Interference: A Journal of Audio Culture*, no. 3, 2013.

[4] S. Waters, "Performance Ecosystems: Ecological approaches to musical interaction," *EMS: Electroacoustic Music Studies Network*, 2007.

[5] P. A. Corning, "The re-emergence of emergence: A venerable concept in search of a theory," *Complexity*, vol. 7, no. 6, pp. 18–30, 2002.

[6] A. Di Scipio, "Listening to yourself through the otherself: on background noise study and other works," *Organised Sound*, vol. 16, no. 02, pp. 97–108, 2011.

[7] H. R. Maturana and F. J. Varela, *Autopoiesis and cognition.* Reidel, Dordrecht, 1980.

[8] G. Bateson, *Steps to an ecology of mind.* University of Chicago Press, 1972.

[9] A. Etxeberria, J. J. Merelo, and A. Moreno, "Studying organisms with basic cognitive capacities in artificial worlds," *Cognitiva*, vol. 3, no. 2, pp. 203–218, 1994.

[10] X. Barandiaran and A. Moreno, "On what makes certain dynamical systems cognitive: A minimally cognitive organization program," *Adaptive Behavior*, vol. 14, no. 2, pp. 171–185, 2006.

[11] R. Benkirane, E. Morin, I. Prigogine, and F. Varela, *La complexité, vertiges et promesses: 18 histoires de sciences.* Le pommier, 2002.

[12] M. Mitchell, "Complex systems: Network thinking," *Artificial Intelligence*, vol. 170, no. 18, pp. 1194–1212, 2006.

[13] O. Bown and A. Martin, "Autonomy in musicgenerating systems," in *Eighth Artificial Intelligence and Interactive Digital Entertainment Conference*, 2012, p. 23.

[14] B. Clarke and M. B. Hansen, *Emergence and embodiment: New essays on second-order systems theory.* Duke University Press, 2009.

[15] D. Sanfilippo and A. Valle, "Feedback systems: An analytical framework," *Computer Music Journal*, vol. 37, no. 2, pp. 12–27, 2013.

[16] K. Karplus and A. Strong, "Digital synthesis of plucked-string and drum timbres," *Computer Music Journal*, vol. 7, no. 2, pp. 43–55, 1983.

[17] P. R. Cook, "A meta-wind-instrument physical model, and a meta-controller for real-time performance control," 1992.

[18] D. Rocchesso and J. O. Smith, "Circulant and elliptic feedback delay networks for artificial reverberation," *Speech and Audio Processing, IEEE Transactions on*, vol. 5, no. 1, pp. 51–63, 1997.

[19] F. Heylighen and C. Joslyn, "Cybernetics and second order cybernetics," *Encyclopedia of physical science & technology*, vol. 4, pp. 155–170, 2001.

[20] H. Von Foerster, *Understanding understanding: Essays on cybernetics and cognition.* Springer Science & Business Media, 2007.

[21] W. Ashby, "Requisite variety and its implications for the control of complex systems," *Cybernetica*, vol. 1, pp. 83–99, 1958.

[22] V. Verfaille, “Effets audionumériques adaptatifs: théorie, mise en œuvre et usage en création musicale numérique.” Ph.D. dissertation, Université de la Méditerranée-Aix-Marseille II, 2003.

[23] D. Arfib, J.-M. Couturier, L. Kessous, and V. Verfaille, “Strategies of mapping between gesture data and synthesis model parameters using perceptual spaces,” *Organised sound*, vol. 7, no. 02, pp. 127–144, 2002.

[24] J. Bullock, “Implementing audio feature extraction in live electronic music,” Ph.D. dissertation, Birmingham City University, 2008.

[25] M. Trayle, “Nature, networks, chamber music,” *Leonardo Music Journal*, pp. 51–53, 1991.

[26] C. Haworth, “Ecosystem or Technical System? Technologically-Mediated Performance and the Music of The Hub,” in *Proc. of the Electroacoustic Music Studies Network Conference*, 2014.

[27] J. Bischoff, R. Gold, and J. Horton, “Music for an interactive network of microcomputers,” *Computer Music Journal*, pp. 24–29, 1978.

[28] T. Ingold, *Being alive: Essays on movement, knowledge and description*. Taylor & Francis, 2011.

[29] A. Di Scipio, “The place and meaning of computing in a sound relationship of man, machines, and environment,” *Array*, pp. 37–52, 2015.

[30] H. H. Rutz, “Making a Space of Algorithmicity,” in *Proc. of xCoAx (Computation Communication Aesthetics and X)*, 2016.

[31] J. Bowers, “Improvising machines: Ethnographically informed design for improvised electro-acoustic music,” 2002.

[32] O. Green, “User serviceable parts: Practice, technology, sociality and method in live electronic musicking,” Ph.D. dissertation, City University London, 2013.

[33] E. Hörl, “A Thousand Ecologies: The Process of Cyberneticization and General Ecology,” in *The Whole Earth. California and the Disappearance of the Outside*, D. Diederichsen and A. Franke, Eds. Sternberg Press, 2013.

[34] S. Emmerson, *Living electronic music*. Ashgate Publishing, Ltd., 2013.

[35] P. Sanden, *Liveness in Modern Music: Musicians, Technology, and the Perception of Performance*. Routledge, 2013.

DESIGN IMPLICATIONS FOR TECHNOLOGY-MEDIATED AUDIENCE PARTICIPATION IN LIVE MUSIC

Oliver Hödl, Geraldine Fitzpatrick and Fares Kayali
Vienna University of Technology (TU Wien)
Human Computer Interaction Group
oliver.hoedl@igw.tuwien.ac.at
geraldine.fitzpatrick@tuwien.ac.at
fares@igw.tuwien.ac.at

Simon Holland
The Open University
Music Computing Lab
simon.holland@open.ac.uk

ABSTRACT

Mobile and sensor-based technologies have created new interaction design possibilities for technology-mediated audience participation in live music performance. However, there is little if any work in the literature that systematically identifies and characterises design issues emerging from this novel class of multi-dimensional interactive performance systems. As an early contribution towards addressing this gap in knowledge, we present the analysis of a detailed survey of technology-mediated audience participation in live music, from the perspective of two key stakeholder groups - musicians and audiences. Results from the survey of over two hundred spectators and musicians are presented, along with descriptive analysis and discussion. These results are used to identify emerging design issues, such as expressiveness, communication and appropriateness. Implications for interaction design are considered. While this study focuses on musicians and audiences, lessons are noted for diverse stakeholders, including composers, performers, interaction designers, media artists and engineers.

1. INTRODUCTION

Mobile, ubiquitous, and sensor based technologies have created new possibilities for interactive performance systems in live music. Approaches to design interactive systems particularly for audience participation are diverse and have implications for art, technology, and science [1–10]. Most published studies present systems and their evaluation in the context of interactive live performances, but lack any substantial analysis of general design implications beyond specific use cases.

This paper explores more generally the design space of technology-mediated audience participation, which we abbreviate throughout this paper as *TMAP*. Implications are considered from a range of stakeholder perspectives.

To investigate this hypothesis and identify potential design implications for TMAP, the present study focuses on stakeholder perspectives around motivation, behaviour, and opinion of spectators and musicians in relation to live music and TMAP. The following four research questions are considered:

1. What are the musical preferences, motivations, and behavioural tendencies of spectators and musicians in live concerts?
2. How do spectators and musicians use mobile technology during live concerts?
3. What are the concerns of spectators and musicians regarding TMAP?
4. What implications for the design of TMAP can be identified?

Addressing these questions on the basis of the survey results has the potential to contribute to new knowledge in two ways. Firstly, by the identification and characterisation of design issues in this emerging area of interaction design, and secondly, by encouraging a focus by designers on underexplored strategies and areas of attention within technology-mediated audience participation.

2. RELATED WORK

Audience participation using technically mediated systems has been conducted in a variety of ways. In an early work in 1977, *Radio Net* [1] asked thousands of people all over the United States to use their telephones in a networked performance to create sounds for a live radio broadcast. Around 25 years later, Levin [11] used an audience's mobile phones to collaboratively create the concert *Dialtones*. Both examples were primarily intended as works of art to stand for themselves.

More recently, mobile devices [2, 4], smartphones in particular [5, 7, 8, 10], and other sensory mechanisms [3, 9] have been used to let spectators participate in performances. In some, but not all of those studies, the researchers were interested in gathering feedback from the audience about their experience.

In two studies [2, 4] the audience was surveyed after the performance and gave feedback about technical issues with the system such as responsiveness and latency. By contrast, in the case of Lee et al. [7] spectators were asked to report back on experiential and aesthetic issues, for example respondents noted that they "felt connected to the music and other musicians" (p.454). In all of these cases, however, the feedback was focused on particular aspects of the provided system for audience participation or the specific performance concept.

In contrast, Mazzanti et al. [8] propose six metrics to describe and evaluate concepts for participatory performances at a more general level. They directly address aspects of participatory performances conceptually and technically, although, still tied closely to their particular Augmented Stage platform and to participants' feedback collected during evaluation.

With the piece *Experimence* [12], we followed a different approach and composed a song having audience participation in mind. We reflected on the creative process of this composition and concluded with rather general variables. These variables describe considerations for a composition before having any particular technology for such an interactive performance system available.

In most published studies, participants' feedback about technology-mediated audience participation (TMAP) across different studies is generally positive [2–4, 6, 7]. Feedback in such studies tends to be limited to particular technical systems and tends to be focused on details of interaction modalities and desired improved or additional features. Audiences often express a wish for more control [3, 6]. Musicians, however, appear to be far more sceptical towards new ways of audience participation [6]. However, the literature does not contain much evidence or discussion about these concerns on the musicians' side.

Overall, musicians and audiences have distinctive requirements, as does musical coherence, and there can be wide variation among both groups. As examples in literature suggest, the effective design of TMAP generally requires balancing knowledge from diverse perspectives and taking into account requirements of different roles in live music performance.

The present study is unusual in surveying these requirements from two different perspectives and without any particular TMAP concept or technology in mind. The aim was to identify general design implications as well as potential design strategies for future case studies.

3. SURVEY

The survey was designed to be conducted online using the free open source software LimeSurvey [1]. Participants could choose between a German or an English version.

3.1 Questionnaire Design

The questionnaire begun with questions about basic information, followed by questions about music-related information in general and live music in particular, and finally focusing audience participation in live music. We decided to use these four sections to guide the participant through the survey step by step from very general questions to very particular ones concerning audience participation.

In the second part of the questionnaire the participants had to rate various statements from their point of view and experience using different Lickert scales [13, 14]. These statements were primarily informed by literature investigating the experiences of spectators attending musical live performances [6, 15]. The original questionnaire as well as detailed survey results are available in Appendix B in [16].

Question	Spectators	Musicians
Age (younger than 29)	59%	47%
Gender (male)	57%	80%
Education (college or higher)	43%	70%
Playing an instrument or vocal training (yes)	75%	100%
Attending or playing concerts (once/month or more)	52%	21%

Table 1. Demographics

3.2 Analysis Approach

For the analysis of the results, descriptive statistics and quantitative methods were used. Ways of presenting these results include bar charts showing frequencies of responses as percentages of the whole sample [17]. This analysis approach concerned mainly results presented in 4.1 and 4.4. For questions which allowed a wide range of response options, statistical measures of central tendency [2] were calculated for easier interpretation [18]. Results of these questions are presented in 4.2, 4.3, and 4.5.

4. RESULTS

The survey was carried out online over a period of three weeks resulting in 254 responses. For the analysis, incomplete responses (27) were excluded, which left 227 complete datasets (169 spectators, 58 musicians).

Different channels, mainly Austrian and British, were used to distribute the survey link. Among these channels were mailing lists of universities, music-related projects and communities, personal contacts of involved researchers, and social media. Furthermore, a distribution by companies in the music business (e.g. labels, concert organiser), music related magazines and broadcasting stations was requested. These inquiries mainly remained unconfirmed, though.

4.1 Demographics and Music-Related Information

Musicians and spectators were separately analysed. Table 1 gives a demographic overview of the dataset. In both target groups about half were younger than 29 and there was a good balance among spectators between male and female, while the musicians were predominantly male. Three quarters of all spectators (75%) played instruments or had vocal training. The musicians were not explicitly asked, whether they had a musical training or not. This was assumed, based on their decision to fill out the survey as a musician.

[1] https://www.limesurvey.org (last access 15.05.2017)

[2] median (Md), mode (Mo), and interquartile range (IQR)

4.2 Motivation

The motivational intentions of spectators and musicians show some interesting differences when agreeing or disagreeing to particular statements. Most spectators agreed, that they want to have a unique and special experience (89%), that they like to be part of an audience (83%), and that they think live music is better than listening (81%). Most spectators disagreed with being involved in the show (49%), but also 32% are neutral about this involvement. Furthermore, many spectators agree to meet other people at concerts (78%) and express themselves to show excitement (71%). For musicians it is very important to be on stage (91%) and play music publicly (90%). Most of them also want to create a unique and special experience (84%). The latter is a statement that no musician disagreed with. 40% of the musicians agree to involved spectators in the show. The statements with the most negative responses are focus on show (41%) and improvise on stage (29%).

4.3 Behaviour

To study the behaviour of spectators and musicians during two different kind of songs, survey participants had to rate 14 statements according to how often they see themselves acting alike during a concert. The scale for this rating was 0 = never, 1 = sometimes, 2 = often, and 3 = every time.

According to these numbers, spectators are likely to close their eyes and stand still, listening to the music carefully during a slow song but never during a fast song. Most musicians often close their eyes during certain parts of a song and stand still to enjoy the playing during a slow and sometimes during a fast song.

Behaviour most spectators often show is clapping hands, waving hands in the air, singing along, moving and dancing, and tapping the beat with the foot. A behaviour that most people never do regarding both song types are moshing around in the mosh pit, grabbing a lighter and waving it in the air, and using a camera or phone. Most spectators never use any kind of devices and even the distribution of values is small.

The interquartile ranges calculated for the rating results show that there is a wider distribution of values with statements of what most spectators do every time (e.g. sing along) or what they never do (e.g. shout/whistle). In addition, the modes do not equal the medians.

Musicians also rated how they see themselves playing the two different kinds of songs on stage. Musicians often close their eyes at certain parts of the song, stand still, enjoy playing, and make announcements before/after song when playing a slow song. For a fast song the behaviour is different. Most musicians often watch the reaction of the audience while playing, smile at certain spectators, and move around on stage. Three statements rated by the musicians, “make announcements before/after song”, “watch reaction of audience while playing”, and “smile at certain spectators”, are amongst those rated highest for both song types. The majority of the musicians does most of these actions often.

Frequency	Mobile phone	Smart phone
More than 10 times	2%	4%
4-9 times	2%	8%
1-3 times	32%	53%
never	62%	30%
no answer	2%	5%

Table 2. Spectators’ phone use during concerts

Purpose	Musician use
recording by yourself	24%
creating visuals or projections	3%
displaying something (e.g. lyrics, musical score)	17%
a device for playback reasons	17%
an instrument to play with	12%

Table 3. Musicians’ use of mobile computer devices on stage

4.4 Mobile Technologies

The use of mobile technologies during live concerts was the next part of this survey. Table 2 summarises the frequency of the spectators’ use of mobile phones. According to the results, most spectators use their mobile phones rarely or never during performances.

Spectators use their phones for diverse reasons during a concert. Most of them never use their phones expect for making pictures. The low interquartile range for most purposes indicates that at least some spectators use their mobile phones sometimes for different purposes.

Musicians were not only asked about their phones, but how often they use them and if they also use other mobile computer devices for their performances. Slightly more than a quarter (28%) already used a smart phone during a performance and 7% does so every concert. More than a third (38%), have used laptops and a tenth (10%) use them for every performance. Tablet computers on stage were used at least once by 12%. Particular purposes musicians use their mobile devices for are listed in Table 3.

4.5 Opinion about TMAP

The last survey part asked the participants about their opinion on TMAP. As in previous sections, survey participants again had to rate whether they agree or disagree with different statements using a five-step scale. The first series of statements was formulated as “I would like to influence...” for spectators and “The audience could influence...” for musicians.

Overall, most spectators tend to agree more on influencing elements of sound (e.g. volume) or dramaturgy (e.g. song selection) in a live concert. Most musicians tend to agree on letting the audience participate in visuals (e.g. lights) or dramaturgy as well, but strongly disagree on an influence of sound.

Most spectators tend to agree with having a certain influence on the general volume or the volume of certain instruments referring to sound and the choice of songs in the category dramaturgy (Md=3/Mo=3; 3 is 'tend to agree'). The statement "volume of certain instruments" even has an interquartile range of 1, which indicates a lower distribution and a more stable tend to agree. In most cases spectators have a neutral opinion. Musicians have much stronger opinion. Most of them tend to agree with being able to influence visuals or dramaturgical elements. Except of "general volume" most disagree on all statements within the category sound.

Finally, survey participants had to rate statements about how TMAP could actually work. These statements included general strategies to involve the audience, concrete examples for participation and actual technologies that might be used.

Among the group of spectators is one statement that clearly stands out, "the artist meets the expectations of the audience" as most spectators strongly agree with it (Md=4/Mo=4) and the values are not much distributed (IQR=1). Most spectators could also imagine using phonometers to measure the noise level and making a certain creative contribution. With sensor technology, they could imagine cameras for visual recognition and floor sensors. Most spectators disagree with voting, controlling sound or visuals actively, or providing personal data (e.g. heart rate). In most cases, musicians have a similar opinion as the spectators or they agree even stronger than audience members. With active sound control or sensor data, however, they strongly disagree.

5. DISCUSSION

The survey presented in this paper explores the design space of technology-mediated audience participation (TMAP) in live music from the perspective of two key stakeholder groups (musicians and audiences). We continue to discuss the previously described survey results and revisit the four research questions. By answering the first three research questions and discussing notable tendencies of the results, we will identify implications that concern the design of TMAP.

In particular, we look step by step at noticeable differences of statistical values in the results and draw conclusions in relation to TMAP. We will finish the discussion by revisiting the fourth research question and take the outcomes of the whole survey into consideration to elaborate and propose implications for design of TMAP. To start the discussion, we address the first research question: What are the musical preferences, motivations, and behavioural tendencies of spectators and musicians in live concerts?

5.1 Music-Related Information

Looking at the musical training of spectators, three quarters stated they play instruments. This number is relatively high, bearing in mind that those who filled out the survey as spectators do not consider themselves as musicians. The fact that 75% of the spectators have musical training supports the responses' credibility of this survey regarding music-related questions to spectators. Furthermore, these numbers highlight the issue of musically trained spectators among the audience, or more general, to consider possible skills among the audience for the design of TMAP. We refer to this as *skilfulness*.

A little more than half of the surveyed spectators attend live concerts at least once a month, which is a good amount of people regularly experiencing live music. In the case of musicians only a fifth plays concerts with the same regularity. If we invert this number, it means the majority of the musicians play live concerts less than once a month. This is not as high as one could think of, when asking people who consider themselves as musicians. A possible explanation could be that a certain number of musicians have above-average experiences and regularly play live concerts, but do not make a living out of music. Following this assumption, the aforementioned one fifth could be considered as professionals, which seems to be appropriate for someone who plays a concert every month or even more often. For the design of TMAP this means it is important to consider the professional level of musicians and their live performances. In conclusion, we refer to this as *masterfulness*.

5.2 Motivation to Play or Attend Live Concerts

The participants' motivation in relation to live concerts showed strong agreements in terms of having a distinctive experience. For spectators the strongest motivation for visiting concerts is to have a unique and special experience. Similarly, most musicians want to create a unique and special experience. Additionally, spectators agreed, that live music is better than listening to records. This raises the implication of *distinctiveness*. It refers to the distinctive experience TMAP should create in a live concert.

Following the previous implication, this suggests that TMAP should always create a distinctive experience. However, at the same time only few spectators agreed to be involved in a show. Musicians on the other hand are more prepared to involve the audience but this is still the second lowest among their ratings. In addition, many spectators want to focus on music without distraction, while most musicians agree on focusing on playing music. Strictly speaking, we can interpret this as an indication that people are not really interested in TMAP. Although, it could also mean that we should focus on a well-considered and unobtrusive involvement of spectators and musicians when utilising TMAP to create a unique live music experience for everybody involved. This highlights an implication for the design of TMAP we call *obtrusiveness*.

Two other statements sharing high agreement among the spectators are about being part of an audience and expressing themselves to show excitement. Again, having TMAP in mind, this indicates the importance of the spectators being able to act expressively and identify themselves with the whole audience. This leads to *expressiveness*, which means the design of TMAP needs to consider forms of interaction that enable the spectators to be expressive, whether as individuals, in smaller groups, or as a whole audience. In parallel, Mazzanti et al. [8] refer to expressiveness with their dimensions 'Active/Passive Audience Affinity' and 'Audience Interaction Transparency' to some extent.

The social context was also identified as important. We interpret the high agreement of spectators to meet other people as *sociability*. This sociability refers to social aspects in relation to TMAP. For example, there could be a certain social motivation to enable TMAP from the side of the artist or to participate as a spectator. Either way, it allows spectators to socialise to some extent, whether this is with friends or meeting other people.

Spectators want to have a unique experience and prefer live performed music. Most important for musicians is to be on stage and play music publicly. This very high agreement among musicians to be exposed on stage and the wish for liveness among the audience raises the implication of *exposure*. This not only highlights the importance for musicians to be on stage, but also the need to design TMAP in a way that considers the exposed situation of musicians.

5.3 Behaviour at Live Concerts

Behaviour at live concerts for both spectators and musicians is related to the expressiveness, as discussed above. However, we can see certain differences in behaviour for different song types. This shows that the spectators' and musicians' behaviour depends on songs and the mood they create among spectators, and this raised the implication of *mood*. The challenge for the design of TMAP is to consider the mood and the resulting behaviour of a participating audience as well as the musicians. For instance, some audiences might have a certain mood-driven behaviour a priori (e.g. according to a style of music) and others might change their behaviour according to a particular song that creates a different mood (e.g. the hit of a band versus a new song no one knows).

The wide distribution of values with statements of what spectators do every time (sing along) or what they never do (shout/whistle), supports the assumption of a higher distribution of ratings for these statements. This means that although most spectators always sing along, there is a certain number of spectators who will not always do it. We call these anticipated differences regarding the behaviour among spectators *diversity*.

It is clearer however that spectators do not tend to use objects during a concerts. Three statements are among the four lowest rated ones (using a lighter, phone or camera). In addition, these three are the only ones among all behaviours requiring an object or specific thing. This raises the implication of *objects*, which means we need to consider the role of tangible interfaces in the interaction design for TMAP.

Musicians show that they like to communicate with spectators whether passively, when smiling at them or watching their reaction, or actively, when making announcements. We argue that all of them show some sort of appreciation to the audience. While making announcements has also an informational purpose, the other two show that most musicians care about their spectators, whether by just observing them to see a reaction or actively smiling at them. As we know from Lee at al. [7], spectators report feeling "connected" (p.454) when experiencing TMAP. We call this the implication of *communication*. When TMAP actually happens during a live concert, it most likely needs some sort of communication, whether it is done by the musicians themselves, by a moderator, or in a self-explanatory way.

5.4 Mobile Technologies during Live Concerts

The second research question was: How do spectators and musicians use mobile technology during live concerts? With the implication of objects, we already looked on phone use in relation to the behaviour during songs. Additional survey results show that only 56% use their phones at least once during a live concert. The only reason why spectators use their phones in particular is to take pictures. Most spectators never use their mobile phones for any other purpose. In conclusion, we define this as *readiness*. For TMAP this means we need to consider to what extent an audience is ready for a particular participation. This readiness could be in terms of general availability (e.g. having a device such as a mobile phone that is capable of something) or in terms of a certain knowledge or habit (e.g. using the mobile phone for a particular purpose).

In general, musicians do not use mobile technology for their performances often. What we do not learn from the results, are the reasons why most of the musicians do not use mobile technologies for their performances. This might be for practical reasons because they just do not need them for artistic purposes, but it could also be kind of refusal. In conclusion, we raise the implication of *openness*. This openness is somehow similar to readiness but focuses more on the musicians' relation towards technology as an important part of TMAP.

5.5 Opinion About TMAP

With the third research question we asked: What are the concerns of spectators and musicians regarding TMAP? Overall, most spectators tend to agree more on influencing elements of sound (e.g. volume) or dramaturgy (e.g. song selection) in a live concert. Most musicians tend to agree on letting the audience participate in visuals (e.g. lights) or dramaturgy as well, but strongly disagree on an influence of sound. Interpreting the figures, it is noticeable that most spectators do not care too much about visuals, while most musicians would somehow offer them the chance to participate in light effects, for instance. Regarding sound most spectators would like to have some influence, while most musicians do not want the audience to influence sound. In relation to TMAP we call this *appropriateness*. This means the actual impact that happens through the participation on some performance element has to be chosen and designed in a way both spectators and musicians can live with.

From the result we know that most spectators tend to agree on influencing the sound to some extent. When asked, if they could imagine using a smartphone app to control the sound actively, although, most spectators strongly disagreed. In a similar contradictory way, most spectators tend to agree on influencing the choice of songs, but strongly disagree on using a smartphone app for voting. This inconsistency raises the implication of *contradiction*, which describes the challenge to find a compromise to resolve a contradictory situation. In the same way other studies report that audiences wish to have more influence when experiencing TMAP while musicians tend to be sceptical towards giving them more control [3, 6].

Most spectators and musicians tend to agree that it would make a live concert more exciting if the audience could make a certain creative contribution. Musicians agree on that even more strongly. We do not know which form of creativity musicians had in mind when they rated the statements. If we consider sound as one of the most important creativity-related aspects of a live concert, it contradicts the musicians' refusal of an impact on sound, as we already know. In conclusion, we define the implication of *creativity* dealing with the challenge to what extent TMAP is or has to be a creative contribution to a live concert.

5.6 Implications for Design of TMAP

The final research question builds up on the previous three to draw conclusions: What implications for the design of TMAP can be identified? By answering the previous three research questions and discussing the results of all survey sections step by step, we identified 16 implications concerning the design of TMAP in live music. Some of them primarily address either spectators' or musicians' requirements, others concern both.

In some cases, we identified these implications by drawing together notable results of the survey questions and sections and by considering spectators' as well as musicians' motivation, behaviour, and opinion. Hence, some implications might overlap to some extent, e.g. *readiness* and *openness*. Both address attitudes and habits of the spectators and musicians in relation to technology that might have an impact on the design of TMAP. Nonetheless, readiness highlights more the technological availability and habits of spectators, while openness rather bears the musicians' relation towards technology.

Besides readiness and openness, especially *obtrusiveness* showed a potential scepticism in relation to TMAP on both sides. Although, various case studies in literature [2–4, 6, 7] report mostly positive feedback from participants who really experienced TMAP at live concerts. This indicates a certain difficulty to envision TMAP without experiencing it as our survey participants did.

From a structural point of view, these implications stand by themselves rather than being a complete set of design strategies. However, they are an important step to generalise design strategies around TMAP and to serve as a well-founded starting point for actual design processes. Furthermore, these implications for the design of TMAP complement the range of design implications derived from particular case studies as the ones presented during the discussion of related work.

5.7 Limitations

The survey presented in this paper investigates a diverse range of aspects that concern the design of TMAP. While the results we discussed mainly address a series of general design implications beyond specific use cases, we do not know much about demographic data, for which population the results are representative, and if study participants had prior experience with TMAP. It can be difficult to imagine the implications of TMAP for those who have never experienced such an interactive performance. This is a limitation in terms of what we have learned about the participants' opinion about TMAP as we cannot judge if people answered based on their expectations or prior experience.

Although, all design implications are derived from the survey results in the same way, they address the design of TMAP on different levels. Some of them are more obvious for design (e.g. distinctiveness, sociability, expressiveness), while others are more difficult to consider and orient to (e.g. mood, openness).

6. CONCLUSION

In the existing literature, identification of issues for the design of technology-mediated audience participation (TMAP) in live music has been mostly based on concrete case studies and derived from case-specific feedback of participants. This has resulted in an identifiable gap in design knowledge. Consequently, we conducted an online survey to collect quantitative data about live music and TMAP on a more general basis, detached from any particular case study. The results were based on 227 complete responses of spectators and musicians, analysed through the use of descriptive statistics. With a step by step discussion of these results across survey sections and both target groups, we identified 16 key issues for the design of TMAP in live music. These issues are skilfulness, expressiveness, diversity, objects, readiness, masterfulness, exposure, communication, openness, creativity, distinctiveness, obtrusiveness, sociability, mood, appropriateness, and contradiction.

The most desired idea by spectators was to select songs played during a concert by using TMAP. Visuals in general are sometimes offered by musicians as an element for control by TMAP, but do not concern audience members so much. In the case of sound, audience members mostly wish to control the volume of the music, whereas the musicians mostly reject any influence on sound.

Finally, there are preferences about particular technologies for audience participation. Candidate technologies include recognition systems such as cameras, floor sensors, and phonometers. In the case of smartphone technologies in particular, opinions are divided.

6.1 Future Work

Throughout the discussion, we found additional possible design directions to be investigated in further studies. These rather concrete design ideas concern the possible impact on performance elements and technological preferences. In general, musicians agree with a creative contribution from the audience. We cannot draw further conclusions on the results about how this creative contribution might look. Thus, it should be a potential focus for further studies.

As mentioned in the limitations section earlier, little demographic information is known about the sample of survey participants. This could be a good starting point for further studies to investigate specific target demographic groups (e.g. based on knowledge, experience, or genre).

Methodologically, future work could explore people's opinion by using qualitative methods. In this way, we could better understand why some ideas for TMAP are preferred and others are not.

7. REFERENCES

[1] M. Neuhaus, "The Broadcast Works and Audium," *Zeitgleich: The Symposium, the Seminar, the Exhibition*, pp. 1–19, 1994.

[2] G. McAllister and M. Alcorn, "Interactive performance with wireless PDAs," in *Proceedings of the International Computer Music Conference*, 2004, pp. 1–4.

[3] J. Freeman, "Large Audience Participation, Technology, and Orchestral Performance," in *Proceedings of the International Computer Music Conference*, 2005, pp. 757–760.

[4] M. Feldmeier and J. Paradiso, "An Interactive Music Environment for Large Groups with Giveaway Wireless Motion Sensors," *Computer Music Journal*, vol. 31, no. 1, pp. 50–67, 2007.

[5] N. Weitzner, J. Freeman, S. Garrett, and Y.-L. Chen, "massMobile - an Audience Participation Framework," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, University of Michigan, Ann Arbor, 2012, pp. 1–4.

[6] O. Hödl, F. Kayali, and G. Fitzpatrick, "Designing Interactive Audience Participation Using Smart Phones in a Musical Performance," in *Proceedings of the International Computer Music Conference*, 2012, pp. 236–241.

[7] S. Lee and J. Freeman, "echobo: A Mobile Music Instrument Designed for Audience To Play," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, KAIST, Daejeon, Korea, 2013, pp. 450–455.

[8] D. Mazzanti, V. Zappi, D. Caldwell, and A. Brogni, "Augmented Stage for Participatory Performances," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, London, UK, 2014, pp. 29–34.

[9] B. V. Hout, M. Funk, L. Giacolini, J. Frens, and B. Hengeveld, "Experio: a Design for Novel Audience Participation in Club Settings," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, Goldsmith University, London, UK, 2014, pp. 46–49.

[10] K. Hayes, M. Barthet, Y. Wu, L. Zhang, and N. Bryan-Kinns, "A Participatory Live Music Performance with the Open Symphony System," in *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems - CHI EA '16*. New York, New York, USA: ACM Press, 2016, pp. 313–316.

[11] G. Levin, "Dialtones (A Telesymphony)," 2001. [Online]. Available: http://www.flong.com/projects/telesymphony/

[12] O. Hödl, G. Fitzpatrick, and S. Holland, "Experimence: Considerations for Composing a Rock Song for Interactive Audience Participation," in *Proceedings of ICMC*, 2014, pp. 169–176.

[13] P. H. Rossi, J. D. Wright, and A. B. Anderson, *Handbook of Survey Research*. New York: Academic Press, 1983.

[14] R. Porst, *Fragebogen - Ein Arbeitsbuch*. VS Verlag für Sozialwissenschaften; Auflage: 3, 2011.

[15] K. Burland and S. Pitts, *Coughing and Clapping: Investigating Audience Experience*, K. Burland and S. Pitts, Eds. Surrey, Ashgate, 2014.

[16] O. Hödl, "The Design of Technology-Mediated Audience Participation in Live Music," PhD Thesis, Vienna University of Technology, 2016. [Online]. Available: http://repositum.tuwien.ac.at/obvutwhs/download/pdf/1375209

[17] N. Blaikie, *Analyzing Quantitative Data: From Description to Explanation*. London: Sage Publications Ltd, 2003.

[18] S. Jamieson, "Likert scales: how to (ab) use them," *Medical Education*, vol. 38, no. 12, pp. 1217–1218, 2004.

AUTOMATIC ARRANGING MUSICAL SCORE FOR PIANO USING IMPORTANT MUSICAL ELEMENTS

Hirofumi Takamori
Waseda University
tkmrkc1290@gmail.com

Haruki Sato
Waseda University
haru.freiheit.9@gmail.com

Takayuki Nakatsuka
Waseda University
t59nakatsuka@gmail.com

Shigeo Morishima
Waseda Research Institute of Science and Technology/JST ACCEL
shigeo@waseda.jp

ABSTRACT

There is a demand for arranging music composed using multiple instruments for a solo piano because there are several pianists who wish to practice playing their favorite songs or music. Generally, the method used for piano arrangement entails reducing original notes to fit on a two-line staff. However, a fundamental solution that improves originality and playability in conjunction with score quality continues to elude approaches proposed by extant studies. Hence, the present study proposes a new approach to arranging a musical score for the piano by using four musical components, namely melody, chords, rhythm, and the number of notes that can be extracted from an original score. The proposed method involves inputting an original score and subsequently generating both right- and left-hand playing parts of piano scores. With respect to the right part, optional notes from a chord were added to the melody. With respect to the left part, appropriate accompaniments were selected from a database comprising pop musical piano scores. The selected accompaniments are considered to correspond to the impression of an original score. High-quality solo piano scores reflecting original characteristics were generated and considered as part of playability.

1. INTRODUCTION

Players often enjoy music while performing an instrument. However, it is typically difficult to play music that is composed using multiple instruments and vocal parts, and this is applicable to orchestral, chamber, and pop music. The main reason is that it is necessary to coordinate several individuals who can play an instrument and to adjust their schedules accordingly. Therefore, several studies have focused on support for solo playing by arranging an original score. Additionally, solo arrangements aid in playing comfortably and determining new aspects of the music. They also increase a player's motivation by playing their favorite music. Conversely, creating of an arranged score involves considerable labor and expertise. Hence, the present study investigates a problem involving an automated arrangement for a solo instrument based on music composed of multiple playing parts. Instruments that can simultaneously play the plurality of notes, such as the piano, guitar, and organ, involves various problems such as expressing impressions of original songs and considering playability. This is because an instrument that can play a plurality of notes has an extended range of expression compared with an instrument that only plays monotonic notes. This is followed by focusing on pop music as it involves music composed using multiple instruments. The study also focuses on a piano arrangement since the piano is one of the most famous instruments that generate a wide range of expressions. Therefore, the present study examines the automatic arrangement of pop music for a solo piano.

Generally, extant studies have discussed automating an arrangement for a piano from music composed of multiple parts. Fujita et al. [1] arranged an ensemble score for a piano by using a melody part and a base part that were extracted from an original score. They output piano scores corresponding to different levels of difficulty based on a user's skill level by considering a maximum musical interval and a minimum key stroke duration. Chiu et al. [2] analyzed the role of each part in an original score. The analysis resulted in generating a solo piano score by considering the preservation of original composition, maximum number of simultaneous key strokes, and maximum musical interval of each hand. Onuma et al. [3] analyzed a process of a piano arrangement used by arrangers, constructed a piano arrangement system based on specific problems that occur in the piano arrangement process, and proposed solutions for the same. When all the parts of an original score are summarized into a two-line staff, the notes leading to a problem in an arrangement are automatically detected and conveyed to a user. The choice of a solution depends on the user. Onuma et al. constructed a solo piano arrangement system corresponding to a user's performance skill. Nakamura et al. [4] considered playability based on a piano fingering model by using merged-output Hidden Markov Model. A continuous flow of notes was considered by the fingering model, and two indices were optimized, namely preservation of an original impression and playability. The problem was solved by considering simultaneous playability as well as a continuous of notes in both hands.

A common vein in previous studies involved reducing and selecting notes from an original score in a solo piano arrangement. Extant studies either used original notes directly or used octave shifts. This method prevents a piano-arranged score from dissonance and preserves an original score's impression. Additionally, they can generate a playable score by considering simultaneous playability and a time sequence of notes when notes are re-

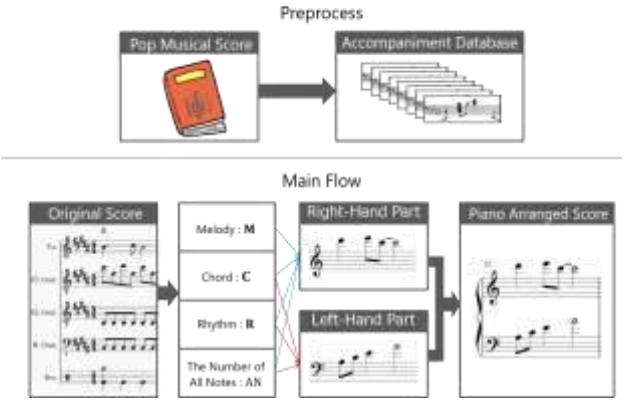


Figure 1. An overview of the proposed method. The preprocess involves constructing an accompaniment database. In the main flow, four musical components are extracted from the original musical score. Subsequently, a piano-arranged score for each hand is separately generated by using musical components. Finally, a correction is performed with respect to overlapped notes.

duced and selected. However, these still remain problems that the outputted piano-arranged score includes difficult notes to perform. A potential reason for this problem is that the definition of playability is not appropriate. Nakamura et al. [4] used a fingering model to consider playability. However, this entails considering several playability factors in addition to the simultaneous maximum number of the key strokes, simultaneous maximum musical interval, minimum duration of time between key strokes such as extension of the maximum musical interval by arpeggio, the cost of fingering, and variation in the difficulty that depends on a tempo. Therefore, it is difficult to define an appropriate restriction with respect to playability. The lack of defining an appropriate restriction of playability can generate a piano score that is not playable in specific places. Extant studies did not discuss the quality of a piano arrangement. However, a piano arrangement is not beautiful if it does not consider a continuous flow of musical sounds such as loudness of sound or change in volume. For example, two phrases, namely “A” and “B”, are involved in a music. The scores that are condensed into two-line staffs from “A” and “B” correspond to the same. Phrase “A” consists of a few instrument parts, and phrase “B” consists of several instrument parts. When piano reduction is considered, the output piano scores are not different between “A” and “B” because both condensed scores created by collecting all notes in the original score are almost the same. Therefore, it is difficult to express a continuous flow of musical sounds. Onuma et al. [3] considered this problem by analyzing a piano arrangement process that is actually performed by arrangers. However, the system is not automatic since the system requires manual corrections by a user.

The present study involves proposing a new approach to arrange a musical score for a piano. Fig. 1 shows an overview of the proposed method. When a piano-arranged score is generated, four important musical components are used, namely a melody, a chord, rhythm, and a number of notes that can be extracted from an original score as opposed to solving the problem of methods to select and delete a large number of notes in an original music score. With respect to the right part, optional notes are added from a chord to the melody. With respect to the left part, appropriate accompaniments are selected from a database that is constructed from pop musical piano scores. The generating process of each playing hand part involves preserving an original score’s impression and improving piano arrangement quality since it is based on an original rhythm and a number of notes for each measure. Rhythm is one of the most important elements of music, and a number of notes are necessary for considering transitions of a song. Additionally, the playability for the left-hand part is improved since it is based on the existing piano score.

2. CLASSIFICATION OF ARRANGING

The word “arranging” has several interpretations. Subsequently, “arranging” is classified into three classes as follows:

(1) Arranging a musical piece that cannot be played by a single instrument into one that can be played by a specific instrument.
(2) Transcribing an atmosphere of a musical piece into mainly a jazz version, a Chopin version and a rock version.
(3) Simplifying an original score which makes it easy to play.

Previous studies examined “arranging” with respect to (2) and (3) ([5-8] and [9-11]). However, there is a paucity of studies examining “arranging” with respect to (1).
The present study focuses on “Arranging” with respect to (1) and setting conditions required involving a high-quality piano arrangement in (1) as shown below:

(i) A melody line is the highest pitch in each measure.
(ii) A chord in a piano-arranged score corresponds to an original one.
(iii) An accompaniment suit for the rhythm part in an original musical piece.
(iv) A piano-arranged score reflects loudness in an original musical piece.
(v) A piano-arranged score is playable.

The reasons for setting the aforementioned five conditions are as follows. (i) The note that corresponds to the highest pitch can correspond to a melody that determines important characteristics of the music because it is more impressive when compared with other notes. Therefore, the impression of a piano-arranged score is significantly different from an original musical piece without condition (i). (ii) Dissonance occurs when a chord in the piano-arranged score does not match an original chord. Subsequently, a piano-arranged score corresponds to an uncomfortable piano-arranged score. Conditions (iii) and (iv)

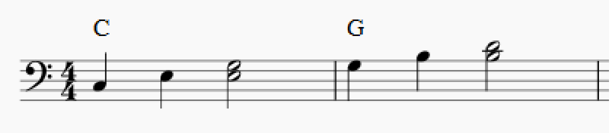


Figure 2. An example of an accompaniment.

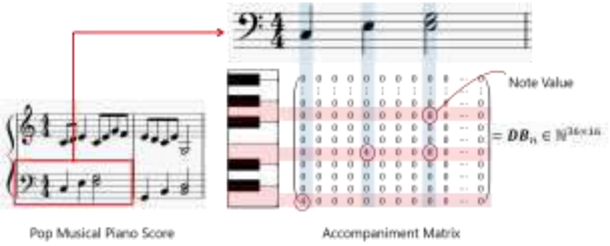


Figure 3. Construction of an accompaniment database.

correspond to the components of improving arrangement quality. (iii) A piano-arranged score becomes a closer impression of an original musical piece by reflecting the rhythm part as an accompaniment. (iv) If a piano-arranged score fulfills condition (iv), then it can express loudness including grand and delicate sounds that change based on the number of sounds of an original musical piece. (v) This condition should be satisfied since the piano-arranged score is finally played by users. In this paper, piano-arranged scores are generated with five conditions.

3. ACCOMPANIMENT DATABASE

Construction of an accompaniment database constitutes a preprocess in the proposed method. This involves considering the manner in which notes information should be treated. Fig. 2 shows an instance of an accompaniment. Each note involves information including sounding start time, pitch, and note value. At the first measure in Fig. 2, a quarter note “C” is put out at the first beat, a quarter note “E” is put out at the second beat, and half notes “E” and “G” are put out at the third beat. Similarly, at the second measure, a quarter note “G” is put out at the first beat, a quarter note “B” is put out at the second beat, and half notes “B” and “D” are put out at the third beat. Both measures completely correspond to the same sounding start times and notes value. The pitches are different in terms of a simple comparison. However, the two measures are the same at a flow of musical sounds, and the impressions generated by them are not different. This implies that the accompaniments that correspond to same sounding start time, notes value, and relative musical intervals between each note and each root note result in a similar impression. In order to reflect this fact, each accompaniment is expressed by a vector of three elements (sounding start time, a relative musical interval based on the root note, and a note value) when an accompaniment database is constructed. Sounding start time at the i-th beat is represented by $4(i-1)$. A half tone is represented by “1”. A whole note is represented by 16. With respect to the above rules of representations, both the measures in Fig. 2 are represented by (0, 0, 4), (4, 4, 4), (8, 4, 8), (8, 7, 8), and they correspond to the same meaning. Subsequently, this vector is rewritten as a matrix that represents relative information including row direction as time, column direction as pitch, and value as note value for each measure. In this study, this matrix is termed as an accompaniment matrix. A left part of a piano-arranged score is created from the accompaniment matrix by providing a root note as absolute information. The process is described in detail in section 5. 2.

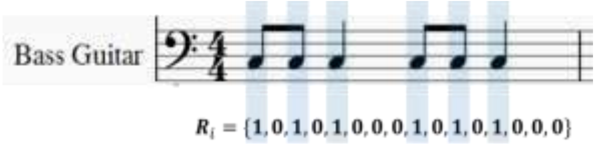


Figure 4. Rhythm extraction from an original music.

Fig. 3 shows the process of constructing an accompaniment database. The left part in a pop musical piano score is divided into a measure. Each measure of an accompaniment is stored as an accompaniment matrix that corresponds to 36 rows and 16 columns. The size of rows corresponds to the number of keys in three octaves, and the size of columns corresponds to the maximum number of notes that can be set notes in a time direction. The creation of an accompaniment matrix involves setting a root note on the bottom row of the accompaniment matrix. For example, a root note at the accompaniment surrounded by a red rectangle line in Fig. 3 corresponds to “C”. The pitch of the first note also corresponds to “C”. The musical interval between the root note and the pitch of the first note corresponds to “0” at the first beat, and thus a note value “4” is set at the bottom left of the accompaniment matrix. The other notes are stored in a manner similar to the first note. In this study, the accompaniment database consists of pop musical piano scores which appear in the website [12]. The accompaniment database involves 285 different patterns in total.

4. EXTRACTION OF MUSICAL COMPONENTS

The proposed method uses four musical components, namely melody **M**, chord **C**, rhythm **R**, and the number of all notes AN. The variables $\mathbf{M}_i$, $\mathbf{C}_i$, $\mathbf{R}_i$, and AN_i represent each musical component with respect to the i-th measure. The melody $\mathbf{M}_i$ is obtained from a vocal score of the original music and is expressed as an 88×16 matrix. The study focuses on a phrase corresponding to a vocal part. A chord is represented by a root note that corresponds to a fundamental note and a chord type that indicates constituent notes of the chord. For example, “CM7” indicates that “C” represents a root note and “M7” represents a type of major 7th. The chord $\mathbf{C}_i$ is obtained from the original chord that is written in the original score. A chord is generally composed in the most popular music genres. This involves measures in which a chord is changing. In order to consider this case, the measure is divided into four blocks, and chords of the original music are assigned to each block of $\mathbf{C}_i$. If there is

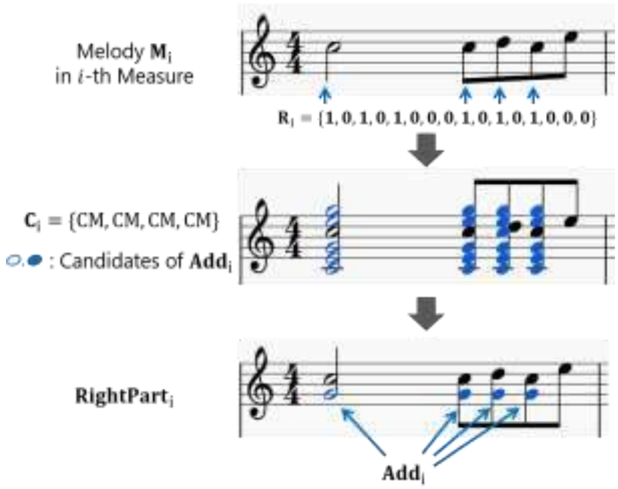

Melody $\mathbf{M}_i$
in i-th Measure
$\mathbf{R}_i = \{1,0,1,0,1,0,0,0,1,0,1,0,1,0,0,0\}$
$\mathbf{C}_i = \{CM, CM, CM, CM\}$
: Candidates of $\mathbf{Add}_i$
$\mathbf{RightPart}_i$
$\mathbf{Add}_i$

Figure 5. The process of generation in the right-hand part.

a music that is not written as chords, then the information in the chords can be extracted by using a web service “Songle” [14]. Rhythm $\mathbf{R}_i$ is extracted from the rhythm part that is designated by a user from an original musical score. Fig. 4 shows the process of rhythm extraction. In this case, the bass guitar is designated as the rhythm part. The rhythm $\mathbf{R}_i$ is represented as a 16-dimension vector. When the i-th measure of the rhythm part is divided into 16 blocks, the integer corresponding to “0” or “1” is stored in each element of $\mathbf{R}_i$. The “0” implies that there are no notes in the block, and “1” implies that there are one or a few notes in the block. The number of all notes AN_i corresponds to the sum of the number of all notes in the i-th measure. The maximum value of AN_i is set as “1” to normalize.

5. GENERATION OF PIANO -ARRANGED SCORE

5.1 Right-Hand Part

Fig. 5 shows the process of generation in the right-hand part. The process is performed with respect to a measure with a note number AN_i that exceeds a threshold $\phi\ (0 < \phi < 1)$ set by a user. This parameter is introduced to provide an impression of thick sound at a measure that involves several notes in the original score. A matrix $\mathbf{RightPart}_i$ is set as the right-hand part in the i-th measure. Thus, $\mathbf{RightPart}_i$ is given as follows:

$$\mathbf{RightPart}_i = \mathbf{M}_i + \mathbf{Add}_i(\mathbf{R}_i, \mathbf{C}_i, \mathrm{AN}_i) \qquad (1)$$

where $\mathbf{Add}_i$ denotes a function that selects an additional note. The algorithm of additional note selection involves three steps. The first step involves obtaining high accent blocks from $\mathbf{R}_i$. The second step involves determining candidates for an additional note to consider a chord $\mathbf{C}_i$ at high accent blocks. The final step involves selecting an additional note that is lower than the melody such that it does not disturb the melody. The operation satisfies conditions (ii) and (iv) that are related to a high-quality piano arrangement as discussed in section 2.

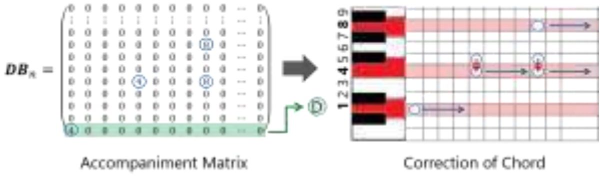


Figure 6. Harmonization process of the generation of the left-hand part from the accompaniment matrix.

5.2 Left-Hand Part

The left-hand part is generated by selection from the accompaniment database based on Dist_i that corresponds to the distance between an original song and arrangement. Following the selection, the selected accompaniment is harmonized based on $\mathbf{C}_i$. A matrix $\mathbf{LeftPart}_i$ is set as the left-hand part in the i-th measure. The left-hand part and distance function are as follows:

$$\mathbf{LeftPart}_i = \mathbf{Hrm}(\arg\min \mathrm{Dist}_i(\mathbf{DB}), \mathbf{C}_i) \qquad (2)$$

$$\mathrm{Dist}_i(\mathbf{DB}_n) = \omega_{\mathrm{AN}} \times |\mathrm{DBAN}_n - \mathrm{AN}_i| + \omega_{\mathrm{R}} \times \sum_{j=1}^{16} |\mathbf{DBR}_{n,j} - \mathbf{R}_{i,j}| \qquad (3)$$

where $\mathbf{Hrm}$ denotes a function of giving original chords, $\mathbf{DB}$ denotes an accompaniment database, $\mathbf{DB}_n$ denotes the n-th accompaniment matrix, and DBAN_n denotes the number of values that are stored as “1” in $\mathbf{DB}_n$. Furthermore, ω_{AN} and ω_{R} represent the weights. The weights are introduced to adjust a balance between the degree of a number of notes and rhythm. Moreover, Dist_i denotes a function that evaluates the similarity between each accompaniment and an original musical score. The maximum value of DBAN_n is set as “1” to normalize. Additionally, $\mathbf{DBR}_n$ corresponds to a rhythm of $\mathbf{DB}_n$ and is represented as a 16-dimension vector in a manner similar to $\mathbf{R}_i$ in section 4. The j-th component of $\mathbf{DBR}_n$ is stored as “1” if there is a number of other than “0” in j-th column of $\mathbf{DB}_n$. It is stored as “0” if the all components in j-th column of $\mathbf{DB}_n$ are “0”. The process selects an accompaniment matrix that minimizes $\mathrm{Dist}_i$and satisfies conditions (ii) and (iii) that relate to a high-quality piano arrangement as discussed in section 2.

The selected accompaniment matrices are represented as relative information. Subsequently, it is necessary to harmonize these accompaniments. Fig. 6 depicts the harmonization process. For example, an accompaniment matrix is output as a chord “Dm” as shown in Fig. 6. First, the bottom row of the matrix is set on root note “D”, and the other components are translated based on the root note in a similar manner. The chord pattern “m” denotes “minor”. When the root note first pitch is set, then “minor” includes the fourth and eighth pitch. However, there are a few notes that are not on the chord "Dm" when notes are simply translated based on the root note. In this case, the notes are relocated to the nearest constituent notes of the chord "Dm".

It is considered that the generated left-hand part score is playable since it comprises an existing piano-arranged score. If the existing piano-arranged score is playable, then the selected accompaniment in the database is also playable. The process of left-hand part generation satisfies condition (v) that relates to a high-quality piano arrangement as discussed in section 2.

5.3 Correction

There is a possibility of overlapping with respect to the notes of the right- and left-hand parts if each part generated in sections 5. 1 and 5. 2 are simply combined. In this case, the problem is solved by reducing either the right-hand or left-hand part. However, the method of reduction results in a few problems. For example, the impression of a piano-arranged score departs from an original score because the accent becomes weaker due to the reduction of notes. Therefore, this problem is solved by relocating an overlapped note in the left-hand part to the nearest constituent note that is lower than the overlapped note. This operation enables the generation of a playable piano-arranged score by preserving an original impression. This process satisfies conditions (ii), (iv), and (v), which are related to a high-quality piano arrangement as discussed in section 2.

6. RESULT AND EVALUATION

Three pop music songs for a solo piano were arranged by using the proposed method, and the quality and playability of the arranged scores were evaluated. The correlation between the value of the distance function Dist_i and subjective evaluation was calculated. This ascertained as to whether the distance function Dist_i is appropriate for reflecting the quality of the piano arrangement. The study also involved investigating as to whether a generated piano score is playable with respect to a piano player's evaluation. The songs used in the study included "*Zenzenzense*" by RADWIMPS, "*Senbonzakura*" by Kurousa-P and "*Sugar Song and Bitter Step*" by UNISON SQUARE GARDEN. Each score as an input was obtained from the website [15], and the base part was set as the rhythm part. Thresholds ϕ was set   and corresponded to 0.86 for "*Zenzenzense*", 0.83 for "*Senbonzakura*", and 0.63 for "*Sugar Song and Bitter Step*".

	Arithmetic mean of ρ	variance of ρ	Subjective evaluation	
			Our method	Arranger
Zenzenzense	0.59	0.19	5.4	5.8
Senbonzakura	0.44	0.39	5.7	—
Sugar Song and Bitter Step	0.34	0.43	5.6	—

Table 1. Spearman's rank correlation coefficient ρ and subjective evaluation value.

Vo.
Cho.
E.G.1
E.G.2
E.P.
Clav.
Synth.
Dr.
Brass
Orch.Hit
St.
Synth.B
Bass

Figure 7. The original score of "RWC-MDB-P-2001 No. 5".

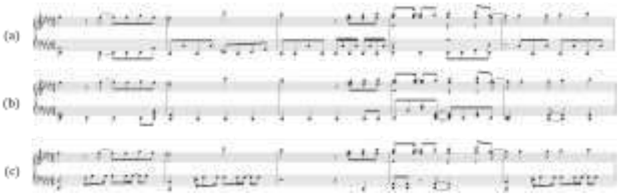


Figure 8. The results of the piano arrangement.

The correlation between the value of Dist_i and subjective evaluation was evaluated by calculating Spearman's rank correlation coefficient. The subjective evaluation was performed by eight evaluators including individuals who can play an instrument. Piano-arranged scores with values of Dist_i corresponding to minimum, median, and maximum were generated. Thus, there are three patterns of piano-arranged scores ranging from the worst arrangement to best arrangement. The evaluators randomly listened to the fore-mentioned three patterns of arrangement quality and evaluated the same by using the following indexes.

1 : extremely low quality.
2 : moderately low quality.
3 : slightly low quality.
4 : neither low nor high quality.
5 : slightly high quality.
6 : moderately high quality.
7 : extremely high quality.

The subjective order was set, and it was sorted in the descending order of indexes set by evaluators. Additionally, the distance values corresponding to the ranking values of Dist_i were set in an ascending order. This is followed by calculating Spearman's rank correlation coefficient ρ between subjective order and distance order. Additionally, the process also involved calculating the arithmetic mean and variance of ρ. Furthermore, "*Zenzenzense*" that corresponds to a piano-arranged score composed by a professional arranger [12] is also evaluated and compared with the best arrangement generated by the proposed method. The result of the evaluation is shown in Tab. 1. With respect to the playability, two individuals who could play the piano evaluated the same by playing the best piano-arranged score in the proposed method. The obtained results corresponded to views that suggested that there are a few difficult points although an unplayable point does not exist.

A part of "RWC-MDB-P-2001 No. 5" recorded in the RWC Music Database is shown as an example of the result [16]. The original score is shown in Fig. 7, and the results of piano arrangement are shown in Fig. 8. The bass part is set as the rhythm part, and the threshold ϕ is set as 0.5. Fig. 8 shows scores (a), (b), and (c) in ascending order of the values of Dist_i.

7. CONSIDERATIONS

The results confirmed that correlations existed between Dist_i and subjective evaluations based on the result of Spearman's rank correlation coefficient. Specifically, with respect to "*Zenzenzense*", the variance is lower than that of the others, and a strong positive correlation exists. It is assumed that rhythm is significantly involved in this result. The piano arrangement of "*Senbonzakura*" involves a rhythm that is a slightly different from that of the original without discomfort since the original rhythm is simple, and thus it is easier to fit any rhythms. Consequently, it is considered that a piano arrangement score that is slightly different from an original score also corresponds to a high-quality piano arrangement. With respect to "*Sugar Song and Bitter Step*", it is difficult to determine an optimum accompaniment from the accompaniment database since the rhythm of the original song is quite complicated. With respect to the rhythm perspective in "*Zenzenzense*", an optimum accompaniment could be selected because the original song of "*Zenzenzense*" involves a moderately complex rhythm. With respect to the calculation of the similarity of rhythm, a more appropriate selection of accompaniment is expected by applying different weights to each beat based on the importance of beats such as strong beats and weak beats. With respect to the quality of a piano arrangement, all songs are evaluated well generally. The piano-arranged score with the proposed method corresponded to approximately 93% of the performance relative to arrangements by a professional arranger. Two evaluators indicated that the piano-arranged score with the proposed method was better than the piano arrangement by a professional arranger. In conclusion, the results confirmed the usefulness of the proposed method.

With respect to the playability, unplayable parts were not revealed since published scores were used for the left-hand part. If the published scores are playable, then a generated left-hand score is also playable. However, the evaluators indicated that there were some parts that were not easy to play. Thus, it was assumed that the right-hand part that includes added notes to the melody is not considered with respect to playability in the proposed method. Therefore, this problem can be solved by considering the continuous of the notes.

8. CONCLUSIONS

In this paper, a piano-arranged score is generated by using four musical components extracted from an original score. Furthermore, the study involved defining five conditions necessary for a high-quality piano arrangement. Subsequently, the piano arrangement process is based on the five conditions. The proposed method involves constructing an accompaniment database in the form of accompaniment matrices. The right-hand part is generated by adding notes of a chord. The left-hand part is generated by selection from the accompaniment database and harmonizing the selected accompaniments. Finally, the right-hand part and the left-hand part are combined with the correction of overlapping notes.

The proposed method highlights a novelty of the piano arranging process. Previous studies used all the information in each part. The generated piano-arranged scores are directly affected by the original score. In contrast, the present study involved generating a piano-arranged score by using musical components that are abstracted from musical scores. Therefore, it is easy to generate a piano-arranged score from acoustic signals. A future study will involve generating a piano-arranged score from original acoustic signals. Additionally, future research will examine the construction of an accompaniment database divided into difficulty levels. It will also explore generating a piano-arranged score with respect to difficulty levels. Furthermore, future study aims also include increasing musical components of a database to select an appropriate accompaniment.

Acknowledgments

This work was supported by JST ACCEL Grant Number JPMJAC1602, Japan.

9. REFERENCES

[1] K. Fujita et al., "A proposal for piano score generation that considers proficiency from multiple part," in Tech. Rep. Special Interest Group on Music and Computer, 2008, pp. 47-52.

[2] C. Shih-Chuan et al., "Automatic system for the arrangement of piano reductions," in Proc. 11th IEEE International Symposium on Multimedia, 2009, pp. 459-464.

[3] S. Onuma et al., "Piano arrangement system based on composers' arrangement processes," in Proc. ICMC, 2010, pp. 191-194.

[4] E. Nakamura et al., "Automatic Piano Reduction from Ensemble Scores Based on Merged-Output Hidden Markov Model," in Proc. ICMC, 2015, pp. 298-305.

[5] K. Oyama et al., “DIVA: An Automatic Music Arrangement Technique Based on Impressions Images,” in J. Society for Art and Science, 2007, pp. 126-135.

[6] Y. Nagahama et al., “Arrangement Software for Video Game Music,” in Tech. Rep. Special Interest Group on Music and Computer, 2013, pp. 1-4.

[7] M. Numao et al., “Constructive Adaptive User Interfaces - Composing Music Based on Human Feelings,” in Tech. Rep. Special Interest Group on Music and Computer, 2001, pp. 49-54.

[8] T. Higuchi et al., “A Bossa Nova Guitar Arrangement System for Given Chord Progression,” in Tech. Rep. Special Interest Group on Music and Computer, 2008, pp. 47-52.

[9] T. Fukuda et al., “The System of Practicing the Piano with Automatic Arranging Function According to User's Technology,” in Proc. 79th National Convention of IPSJ, 2015, pp. 403-404.

[10] C. Oshima et al., “Toward Developing of Gradual Musical Scores that Allow People to Challenge Difficult Pieces,” in Tech. Rep. Special Interest Group on Entertainment Computing, 2006, pp. 57-64.

[11] K. Yoshikawa et al., “Simplification of piano score corresponding to excitement of music,” in Tech. Rep. Special Interest Group on Entertainment Computing, 2006, pp. 57-64.

[12] http://pianobooks.jp/,(accessed 2017-01-31).

[13] Y. Serizawa et al., “A method for extracting melodies distributed to multiple parts,” in Proc. 65th National Convention of IPSJ, 2003, pp. 191-192.

[14] M. Goto et al., “Songle: An Active Music Listening Service Based on Music-understanding Technologies for Audio Signals and Error Corrections by Users,” in J. Information Processing Society Japan, 2013, pp. 1363-1372.

[15] http://guitarlist.net/,(accessed 2017-01-31).

[16] M. Goto et al., “RWC music database: Database of Copyright-cleared Musical Pieces and Instrument Sounds for Research Purposes,” in J. Information Processing Society Japan, 2004, pp. 728-738.

FFT-BASED DYNAMIC RANGE COMPRESSION

Leo McCormack and Vesa Välimäki
Acoustics Lab, Dept. of Signal Processing and Acoustics
Aalto University, FI-02150, Espoo, Finland
leo.mccormack, vesa.valimaki@aalto.fi

ABSTRACT

Many of the dynamic range compressor (DRC) designs that are deployed in the marketplace today are constrained to operate in the time-domain; therefore, they offer only temporally dependent control of the amplitude envelope of a signal. Designs that offer an element of frequency dependency, are often restricted to perform specific tasks intended by the developer. Therefore, in order to realise a more flexible DRC implementation, this paper proposes a generalised time-frequency domain design that accommodates both temporally-dependent and frequency-dependent dynamic range control; for which an FFT-based implementation is also presented. Examples given in this paper reveal how the design can be tailored to perform a variety of tasks, using simple parameter manipulation; such as frequency-depended ducking for automatic-mixing purposes and high-resolution multi-band compression.

1. INTRODUCTION

A dynamic range compressor (DRC) is an indispensable tool found in many an audio engineer's toolbox. They are utilised extensively in the fields of music production, automatic-mixing, mastering, broadcasting and hearing aids. From the music production perspective, the main reason for their popularity lies with their ability to shape the amplitude envelope of transient audio signals; so that they may combine more cohesively with other signals present in the mix, or to rectify issues that an existing mix may exhibit. DRCs are also utilised to intentionally reduce the dynamic range of audio signals, as humans tend to associate a louder mix as being superior [1, 2]; which is especially important given stringent broadcasting regulations and the competition between broadcasters.

While the basic design of time-domain DRCs has evolved slowly over the years [3–8], more progress has been made with regard to diverse and creative ways of applying such devices in the field of music production. For example, most commercial implementations of modern DRCs offer the ability to influence the behaviour of the DRC using a different audio signal; a technique commonly referred to as *side-chain compression* [9]. To give a practical example, side-chain compression can be used to reduce the amplitude of a bass guitar signal to coincide with a kick drum transient. This temporally dependent "ducking" in the bass guitar signal may then allow the two instruments to better complement one another. In the production of dance music, this technique is commonly utilised in the more extreme case, by allowing the kick drum transient to conspicuously carve out temporally dependent *holes* in the amplitude of other signals; resulting in a *pumping* effect that pulses in time with the music [10].

Serial and parallel compression techniques have also become widely used, as they allow for even greater control of the amplitude envelope of signals by utilising multiple DRCs with different envelope detector settings [11, 12]. Recent research has also considered the digital modelling of traditional analogue DRC devices [8], as it is often desirable to replicate traditional audio processing techniques, in order to capture their unique distortion characteristics [13].

However, the majority of today's DRC designs are temporally-dependent and frequency-independent (referring to common designs described in [4, 14, 15]), which makes them unsuitable for certain tasks. For example, it might be desirable for the DRC to only derive its gain parameter from a specific frequency range of the input audio, or the user may want to use different DRC parameters for different frequency bands and have them operate independently. Therefore, to accommodate these scenarios, an element of frequency dependency has been adopted by two main sub-categories of DRC design, termed here as the *filtered side-chain compressor* (FSCC) [9, 16] and the *multi-band compressor* (MBC) [6, 17].

In the case of the FSCC, the amount of gain reduction applied to an input signal is influenced by a filtered version of the input signal. One common implementation is the *De-esser* design, which is typically used to reduce sibilance present in vocal recordings [9]. In this instance the gain factor is derived from the frequency range where the sibilance is most prevalent, through the use of a band-pass filter in their side-chain processing. There are also DRC designs that extend this control further and allow much more customisable filtering of the side-chain signal [18], which are suitable for applications where a single band-pass filter is insufficient. However, such designs do not feature in any formal publications. One potential drawback of the FSCC approach is that the calculated gain factor is applied to the whole frequency range of the input signal, which may not be desirable depending on the use case. Therefore, it might be beneficial if FSCC designs were able to apply the calculated gain factor to a user specified frequency range.

For the MBC approach, the input signal is divided into separate sub-bands and offers independent user controls for each of them. This allows for greater control of broadband signals, which therefore makes them especially useful for mastering applications; however, typically only two or three independent sub-bands are controllable by the user. MBCs with a higher number of sub-bands are used more commonly in hearing-aid designs [19–21]. However, due to their specific intended application, and their general lack of envelope detection or side-chain compression support; they are usually unsuitable for music production, mastering or automatic-mixing applications. Designs that are orientated towards these musical applications, such as the Soniformer audio plug-in [22], do not feature in any formal publications.

With regard to today's automatic-mixing approaches [23–25], many of them operate in the frequency or time-frequency domains and are applied off-line. The latest implementations in particular rely heavily on machine learning principles; therefore, they have removed the audio producer entirely from the mixing process and have generally yielded inferior results, when compared to their professionally mastered counterparts [25]. Improvements could be expected, by utilising higher frequency-resolution MBC designs that apply frequency dependent side-chain ducking. This would allow signals to frequency-dependently attenuate the amplitude of other signals, thus creating "space" for them in the mix, while also keeping the human element in the mixing process.

Therefore, the purpose of this paper is to present a generalised time-frequency domain DRC design, which can be utilised as a more customisable FSCC, an expanded MBC, and a potential semi-automatic-mixing method. The paper also provides details of an implementation of the proposed design, which is based on the Fast Fourier Transform (FFT). As demonstrated in the examples presented, the implementation provides promising results, while using a variety of different audio material and simple parameter manipulation.

This paper is organised as follows: Section 2 provides a short introduction on how DRCs generally operate and also describes the typical parameters that are offered to the user; Section 3 gives an example of a time-domain DRC design, which is then expanded to the proposed time-frequency domain design in Section 4; Section 5 then details how the proposed design was implemented, in order to provide the examples that are shown in Section 6; and Section 7 concludes the paper.

2. BACKGROUND

Typically, DRCs operate by duplicating the input signal to obtain two identical signals, henceforth referred to as the *input signal* and the *side-chain signal*. However, it is important to note that the side-chain signal can be an entirely different signal, in the case of the side-chain compression technique. The output signal is then created by attenuating the input signal based on a gain factor that is calculated from processing the side-chain signal. This side-chain processing typically comprises two main components, a *gain computer* and an *envelope detector* [14]. These two components allow many user parameters to be delivered, which typically include:

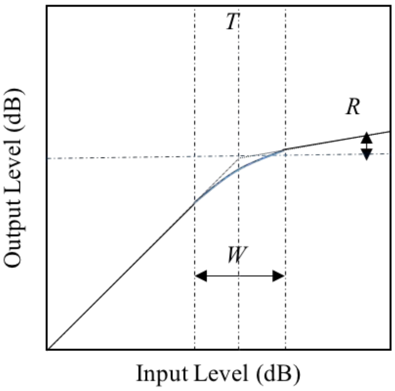


Figure 1: A depiction of the threshold T, ratio R and knee width W parameters. The blue line represents a soft knee, while the black dotted line represents a hard knee.

- *threshold*, above which attenuation of the input signal occurs;
- *ratio*, referring to the input/output ratio in decibels, which influences the gain reduction based on the extent to which the signal has surpassed the threshold;
- *knee width*, which permits a less abrupt attenuation around the threshold by allowing the gain factor to be determined by a gradual change in ratio. This is often termed as a *soft knee*, whereas the abrupt application of attenuation about the threshold is termed as a *hard knee* (see Fig. 1);
- *attack time*, which determines how quickly the envelope detector reaches the target gain reduction, as dictated by the threshold and ratio parameters;
- *release time*, which refers to how quickly the envelope detector returns to unity gain, when the side-chain signal ceases to exceed the threshold.

Additionally, the following parameters can also be offered to the user:

- *look-a-head*, the amount of time the input signal is delayed by, allowing the side-chain processing to react more reliably to acute transients;
- *make-up gain*, which is a gain factor applied after application of the DRC, as it is often desirable to increase the amplitude of the output in order to compensate for the loss in sound energy;
- *side-chain filter parameters*, in order to dictate which frequencies of the side-chain will influence the dynamic range compression of the input signal (used in FSCC designs).

Note that DRCs can either have a feed-forward or feed-back architecture [26]. However, feed-forward designs are generally more preferable, as the feed-back topology prohibits the use of a look-a-head parameter.

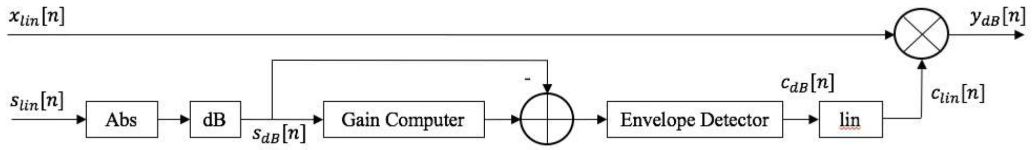


Figure 2: The block diagram of the time-domain feed-forward DRC, derived from [14]. Note that the s_{lin} signal can be identical to the x_{lin} signal, in order to perform typical dynamic range compression behaviour; or alternatively, they can be independent signals, in order to perform the side-chain compression technique.

3. A TYPICAL TIME-DOMAIN DRC DESIGN

An example of a feed-forward time-domain DRC design is described in [14] and features a second order gain computer and a smooth peak envelope detector; the block diagram of which is depicted in Fig. 2. Note that it is recommended to place the envelope detector after the gain computer, in order to mitigate certain artefacts described in [27].

This design operates in the log-domain, as this is a more perceptually relevant and effective domain for applying dynamic range compression

$$s_{dB}[n] = 20 \log_{10} |s_{lin}|, \tag{1}$$

where s_{dB} and s_{lin} refer to the log-domain (decibels) and linear versions of the side-chain signal respectively and $|.|$ refers to the magnitude of a scalar. Alternatively, the root-mean-square (RMS) values of the input signal may be used in situations where the induced latency poses no issue [15].

The gain computer provides one target output gain per sample, $y_G[n]$, and is calculated as

$$y_G[n] = \begin{cases} s_{dB}[n] & 2(s_{dB}[n]-T) \leq -W \\ s_{dB}[n] + \\ \frac{(\frac{1}{R}-1)(s_{dB}[n]-T+\frac{W}{2})^2}{2W} & 2|(s_{dB}[n]-T)| \leq W \\ T + \frac{(s_{dB}[n]-T)}{R} & 2(s_{dB}[n]-T) > W, \end{cases} \tag{2}$$

where T and W are the threshold and knee width parameters, respectively, expressed in decibels and R is the compression ratio.

The envelope detector is then formulated as [28]

$$v_L[n] = \begin{cases} \alpha_A v_L[n-1] + \\ (1-\alpha_A) b_G[n] & b_G[n] > v_L[n-1] \\ \alpha_R v_L[n-1] + \\ (1-\alpha_R) b_G[n] & b_G[n] \leq v_L[n-1], \end{cases} \tag{3}$$

where v_L refers to the output of the envelope detector in decibels; b_G is the input gain, calculated as the difference between the gain computer estimate and the original side-chain value $b_G = y_G - s_{dB}$; and the attack and release time constants are formulated as $\alpha_A = e^{-1/attack}$ and $\alpha_R = e^{-1/release}$ respectively, where the attack and release times are given as samples per millisecond.

The final gain factor c_{lin} is then obtained by returning to the linear domain

$$c_{lin}[n] = 10^{v_L/20}. \tag{4}$$

An example of this DRC design applied to a drum sample, is shown in Fig. 3. It can be observed that the gain reduction is temporally dependent on the transients of the side-chain signal; where transients with larger magnitude have resulted in a more dramatic gain reduction and the envelope detector has influenced the smoothness of the transitions between unity gain and gain reduction.

Note that this design can also be converted to an FSCC design by applying a filter $F(z)$ to the side-chain signal, such as a band-pass filter in the case of a De-esser.

4. THE PROPOSED TIME-FREQUENCY DOMAIN DRC DESIGN

This section details the proposed DRC algorithm, which assumes that the input and side-chain signals have been

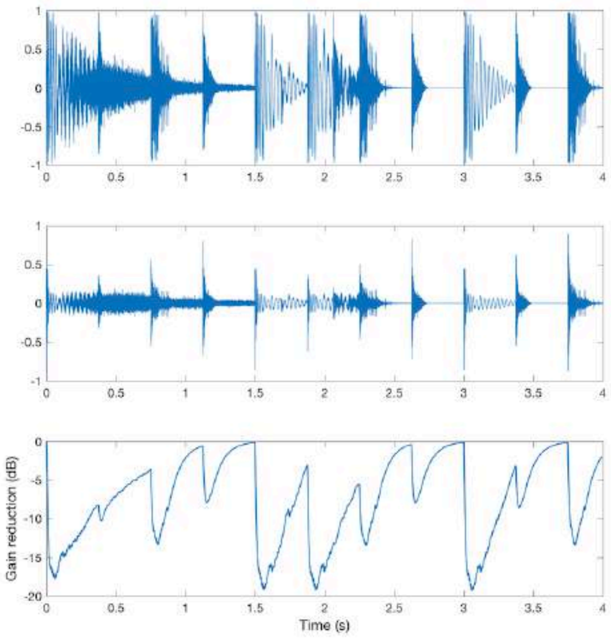


Figure 3: An example of a drum sample (top), after it has been passed through the time-domain DRC design (middle); using $T = -20$ dB, R = 8:1, W = 0 dB, Attack = 10 ms and Release = 80 ms. The gain reduction over time is also depicted (bottom), which shows how the DRC has responded to the transients in the input signal.

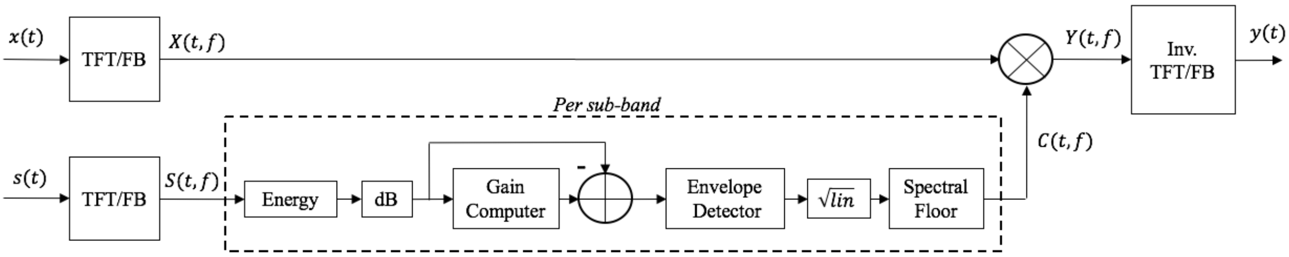


Figure 4: Block diagram of the proposed time-frequency domain DRC design, where TFT/FB denotes a time-frequency transform or perfect reconstruction filterbank. Note, however, that if $x(t)$ and $s(t)$ are the same signal, then the TFT/FB should be applied only once.

transformed into the time-frequency domain, via a Short-Time Fourier Transform (STFT) or a perfect reconstruction filterbank. Therefore, time-domain frames of N input samples $\tilde{x} = [x_0, ..., x_{N-1}]^T$ are now represented as their time-frequency domain counterparts $X(t, f)$; where t refers to the down-sampled time index and $f \in [0, ..., L-1]$ to the frequency indices given by a hop size L.

The proposed time-frequency domain architecture is shown in Fig. 4 and shares many similarities with its time-domain counterpart (see Fig. 2).

However, due to the change in domain, it is now the *energy* of each time-frequency index that is used as the input for the gain computer

$$X_G(t, f) = 10 \log_{10} |S(t, f)|^2, \tag{5}$$

where X_G and S refer to the side-chain energy and the side-chain signal respectively.

For the gain computer, the frequency-dependent energy values are compared against their respective threshold parameters, so that the dynamic range compression may be applied in a frequency-dependent manner (note that the time and frequency indices (t, f) have been omitted for compactness)

$$Y_G = \begin{cases} X_G & 2(X_G - T) \leq -W \\ X_G + \\ \frac{(\frac{1}{R}-1)(X_G - T + \frac{W}{2})^2}{2W} & 2|(X_G - T)| \leq W \\ T + \frac{(X_G - T)}{R} & 2(X_G - T) > W, \end{cases} \tag{6}$$

where Y_G is the new target gain, expressed in decibels; T and W are the threshold and knee width parameters respectively, which are also expressed in decibels and R is the compression ratio.

The envelope detector also works in a similar manner to that of the time-domain implementation in (3)

$$V_G = \begin{cases} \alpha_A V_G z^{-1} + (1 - \alpha_A) B_G & B_G > V_G z^{-1} \\ \alpha_R V_G z^{-1} + (1 - \alpha_R) B_G & B_G \leq V_G z^{-1}, \end{cases} \tag{7}$$

where V_G refers to the output of the envelope detector in decibels and B_G refers to the input gain, calculated as the difference between the gain computer estimate and the energy of the side-chain $B_G = Y_G - X_G$. However, the attack and release time constants must now take into account the change of domain, so are formulated as: $\alpha_A = e^{-1/(attack/L)}$ and $\alpha_R = e^{-1/(release/L)}$ respectively, where the attack and release times are given as time-domain samples per millisecond.

Due to the nature of spectral processing, a spectral floor parameter λ is now recommended to mitigate audible artefacts that can occur when a frequency band is too harshly attenuated. Therefore, the final temporally-dependent and frequency-dependent gain factors $C(t, f)$ are given as

$$C(t, f) = \max\left(\lambda, \sqrt{10^{\frac{V_G(t,f)}{20}}}\right), \tag{8}$$

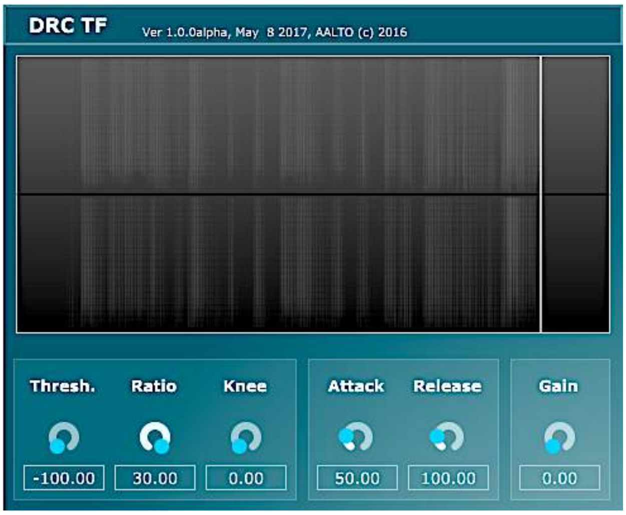
which should be multiplied element-wise with the input signal, in order to yield the dynamically compressed audio signal, which can be auditioned after performing an appropriate inverse time-frequency transform.

5. AN FFT-BASED IMPLEMENTATION OF THE PROPOSED DESIGN

In order to investigate the behaviour and performance of the proposed design, the algorithms presented were implemented using MatLab and then realised as a Virtual Studio Technology (VST) audio plug-in [1]. The graphical user interface (GUI) was designed using the open source JUCE framework and is depicted in Fig. 5.

The alias-free STFT filterbank in [29], which uses analysis and synthesis windows that are optimised to reduce temporal aliasing, was selected as the time-frequency transform for the implementation. A hop size of 128 samples and an FFT length of 1024 samples were selected, with a further sub-division of the lowest four bands to increase low-frequency resolution. This additional filtering is similar to the hybrid filtering that is performed in the filterbanks used for spatial audio coding [30]; therefore, there are 133 frequency bands in total, with centre frequencies that roughly correspond to the resolution of human hearing. A sample rate of 48 kHz and a frame size of 512 samples were hard coded into the plug-in.

[1] Various pre-rendered sound examples and Mac OSX/Windows versions of the VST audio plug-in, can be found on the companion web-page: http://research.spa.aalto.fi/publications/papers/smc17-fft-drc/



Thresh. -100.00
Ratio 30.00
Knee 0.00
Attack 50.00
Release 100.00
Gain 0.00

Figure 5: An image of the GUI for the VST implementation of the proposed algorithm. Note that the same parameters are applied for each sub-band; although frequency-dependent parameter selection is supported internally.

A total of 8 seconds of historic frequency-dependent gain factors $C(t, f)$, utilised in the processing section of the VST, are stored and displayed on the GUI to provide the user with visual feed-back of the extent of gain reduction.

Some of the drawbacks of the proposed design are: the numerous control parameters; increased latency and computational complexity. Therefore, in order to reduce the number of user parameters on the GUI, the same threshold, ratio, knee-width, attack time and release time values are used in the side-chain processing for all of the 133 sub-bands. However, this implementation should be viewed as a demonstration that the proposed algorithm can operate in real time; whereas the MatLab implementation has left all of the user parameters open to manipulation, in order to provide the examples shown in Section 6. With regard to the induced latency, due to the manner in which Digital Audio Workstations (DAWs) operate, delay-compensation can be introduced to make this drawback less of an issue.

6. EXAMPLE APPLICATIONS

The purpose of this section is to provide some perspective on the wide range of tasks that the proposed design can be applied to, via simple manipulation of the calculated gain factors. For example, a standard time-domain DRC design can be modelled by taking the mean of the frequency-dependent gain factors and applying this average to all of the sub-bands equally, which would enable an outcome similar to the example given in Section 3. However, as this results in increased latency and computational complexity, this is not recommended for this particular use case. However, a more appropriate use of this concept is to take the mean of the gain factors for a certain frequency range of the side-chain signal in order to to mimic a customisable FSCC. Now, unlike its time-domain counterpart, the proposed system is capable of applying this average gain factor to a specified frequency-range of the input signal. For example, high frequencies can now be attenuated during periods of high energy in the low frequencies.

The proposed design can also be used simply as an MBC, but with a higher frequency-resolution when compared to most MBCs employed in today's marketplace. The result after passing a drum sample though the proposed design is shown in Fig. 6b Since the individual components of the drum kit (snare, kick drum and hi-hat) are less likely to coincide simultaneously in time, or overlap substantially with regard to their frequency content, the music producer is able to use one instance of the proposed DRC and still dictate different parameters for each component. The alternative would be three independent instances of a time-domain DRC for each drum kit component.

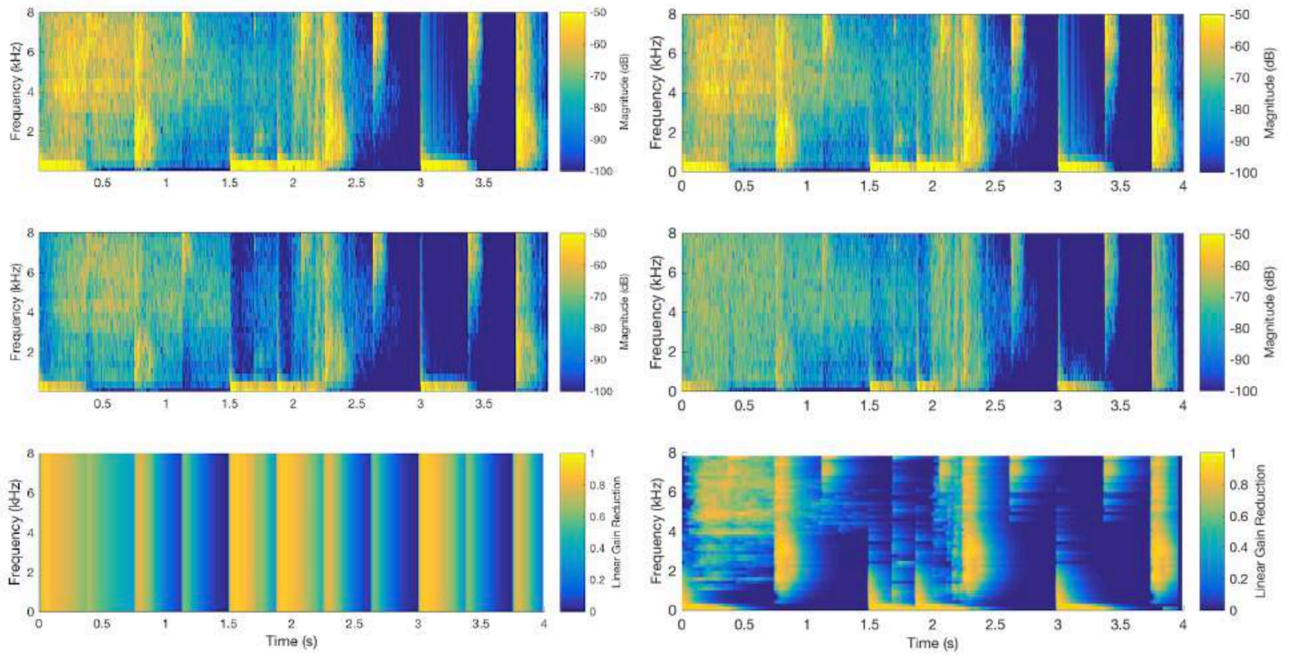
Another potential use for the proposed design is for automatic-mixing purposes. The example shown in Fig. 7 demonstrates how a popular music track, which can be approximated as uniformly distributed white noise, can be subjected to both frequency and temporally dependent side-chain compression, in order to accommodate an additional drum loop sample. One alternative method for attaining this result would be through the use of automation and digital audio filters, which would generally be considered a much more laborious process for accomplishing the same task.

In broadcasting, dynamic range compression is utilised heavily for both limiting and side-chain compression. The former is to ensure that the sound levels do not exceed the limits set by the broadcast regulators and by using the proposed design, only the frequency bands that exceed the limit will be attenuated. The latter is used to decrease the level of audio signals if speech signals exceed a particular threshold; an example utilising the proposed design is shown in Fig. 8.

Please note, however, that the user parameters selected for the presented examples (including Fig. 5) have been purposely configured to more harshly attenuate the signals, in order to make the changes in the spectrogram magnitudes easier to identify. More conservative gain reduction should be used during listening. Likewise, the frequency range shown on the spectrograms has been truncated to 8 kHz (to avoid depicting *dead-space*), despite the processing using a sample rate of 48 kHz.

7. CONCLUSION

This paper has outlined a design for a time-frequency domain DRC, which applies dynamic range compression with both temporally and frequency-dependent characteristics; allowing for increased flexibility when processing audio signals, owing to the greater degree of control that the system provides to the sound engineer. To the best of the authors' knowledge, this approach to dynamic range compression has historically featured mainly in algorithms orientated towards hearing aids, with regard to formal publications. Therefore, some applications have been presented, which highlight the potential usefulness of such a design in the fields of music production, mastering and broadcasting. These include frequency-dependent side-chain ducking for automatic-mixing purposes, and high



(a) $T = -20$ dB, R = 8:1, W = 0 dB, Att/Rel = 10/80 ms.

(b) $T = -50$ dB, R = 8:1, W = 0 dB, Att/Rel = 10/80 ms.

Figure 6: Spectrograms of the drum kit sample (top left, top right) and the output audio, when utilising the time-domain (middle left) and the proposed (middle right) DRC designs; also shown are the frequency-dependent gain factors for the time-domain (bottom left) and the proposed (bottom right) DRC designs.

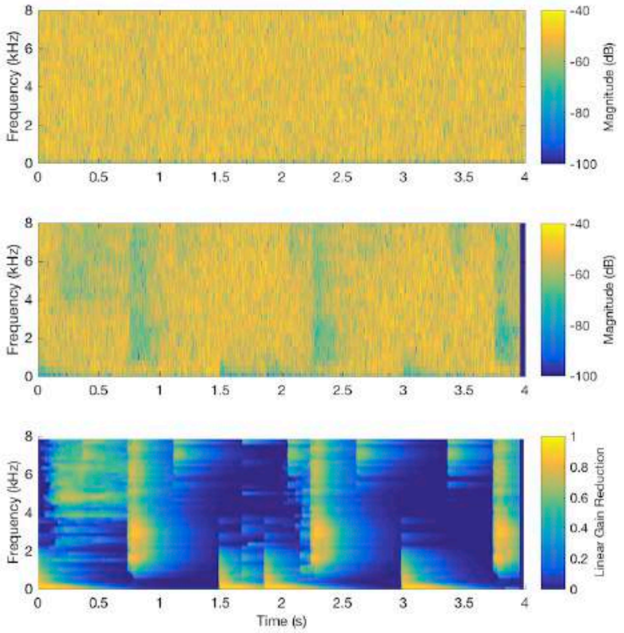


Figure 7: Spectrograms of white noise (top) and after frequency-dependent ducking using a drum kit sample (middle); also shown are the frequency-dependent gain factors (bottom). $T = -50$ dB, R = 8:1, W = 0 dB, Att/Rel = 10/150 ms.

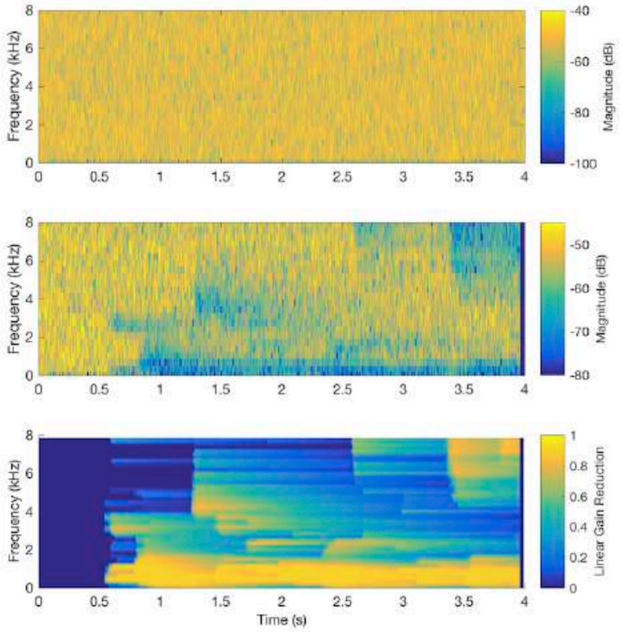


Figure 8: Spectrograms of white noise (top) and after frequency-dependent ducking using a speech sample (middle); also shown are the frequency-dependent gain factors (bottom). $T = -80$ dB, R = 20:1, W = 0 dB, Att/Rel = 10/800 ms.

resolution multi-band compression.

Regarding the presented FFT-based implementation, future work could investigate how best to design a GUI that accommodates the additional parameters, or study how one might automate them; such as in [31].

8. REFERENCES

[1] E. Vickers, "The loudness war: Background, speculation, and recommendations," in *Audio Engineering Society Convention 129*, San Francisco, CA, USA, Nov. 2010.

[2] M. Wendl and H. Lee, "The effect of dynamic range compression on perceived loudness for octave bands of pink noise in relation to crest factor," in *Audio Engineering Society Convention 138*, Warsaw, Poland, May 2015.

[3] B. Blesser, "Audio dynamic range compression for minimum perceived distortion," *IEEE Transactions on Audio and Electroacoustics*, vol. 17, no. 1, pp. 22–32, Mar. 1969.

[4] E. F. Stikvoort, "Digital dynamic range compressor for audio," *Journal of the Audio Engineering Society*, vol. 34, no. 1/2, pp. 3–9, Jan./Feb. 1986.

[5] A. Schneider and J. Hanson, "An adaptive dynamic range controller for digital audio," in *Proc. IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, Victoria, BC, Canada, May 1991, pp. 339–342.

[6] J. C. Schmidt and J. C. Rutledge, "Multichannel dynamic range compression for music signals," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-96)*, vol. 2, Atlanta, GA, USA, May 1996, pp. 1013–1016.

[7] U. Zölzer, *Digital Audio Signal Processing*. John Wiley & Sons, 2008.

[8] P. Raffensperger, "Toward a wave digital filter model of the Fairchild 670 limiter," in *Proc. 15th Int. Conf. Digital Audio Effects (DAFx-12)*, York, UK, Sept. 2012, pp. 195–202.

[9] A. J. Oliveira, "A feedforward side-chain limiter/compressor/de-esser with improved flexibility," *Journal of the Audio Engineering Society*, vol. 37, no. 4, pp. 226–240, Apr. 1989.

[10] Music Radar, "How to create classic pumping sidechain compression," http://www.musicradar.com/tuition/tech/how-to-create-classic-pumping-sidechain-compression, 2015, [Online; accessed 01-Feb-2017].

[11] Sound on Sound, "Parallel compression," http://www.soundonsound.com/techniques/parallel-compression, 2013, [Online; accessed 27-Jan-2017].

[12] MusicTech, "Pro Tools tutorial: Cutting edge production techniques—Serial compression serial compression," http://www.musictech.net/2015/05/pro-tools-serial-compression/, May 2015, [Online; accessed 28-Jan-2017].

[13] A. Moore, R. Till, and J. Wakefield, "An investigation into the sonic signature of three classic dynamic range compressors," in *Audio Engineering Society Convention 140*, Paris, France, May 2016.

[14] D. Giannoulis, M. Massberg, and J. D. Reiss, "Digital dynamic range compressor design: Tutorial and analysis," *Journal of the Audio Engineering Society*, vol. 60, no. 6, pp. 399–408, June 2012.

[15] G. W. McNally, "Dynamic range control of digital audio signals," *Journal of the Audio Engineering Society*, vol. 32, no. 5, pp. 316–327, May 1984.

[16] R. J. Cassidy, "Dynamic range compression of audio signals consistent with recent time-varying loudness models," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, vol. 4, Montreal, QC, Canada, May 2004, pp. 213–216.

[17] S. H. Nielsen and T. Lund, "Level control in digital mastering," in *Audio Engineering Society Convention 107*, New York, USA, Sept. 1999.

[18] Fab Filter, "Manual: Pro MB," http://www.fabfilter.com/help/ffpromb-manual.pdf, 2016, [Online; accessed 31-Jan-2017].

[19] D. K. Bustamante and L. D. Braida, "Multiband compression limiting for hearing-impaired listeners," *Journal of Rehabilitation Research and Development*, vol. 24, no. 4, pp. 149–160, 1987.

[20] B. Kollmeier, J. Peissig, and V. Hohmann, "Real-time multiband dynamic compression and noise reduction for binaural hearing aids," *Journal of Rehabilitation Research and Development*, vol. 30, no. 1, pp. 82–94, 1993.

[21] E. Lindemann, "The continuous frequency dynamic range compressor," in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, NY, USA, Oct. 1997, pp. 1–4.

[22] Voxengo, "Soniformer audio plugin," http://www.voxengo.com/product/soniformer/, 2016, [Online; accessed 31-Jan-2017].

[23] E. Perez-Gonzalez and J. Reiss, "Automatic equalization of multichannel audio using cross-adaptive methods," in *Audio Engineering Society Convention 127*, New York, NY, USA, Oct. 2009.

[24] J. D. Reiss, "Intelligent systems for mixing multichannel audio," in *Proc. IEEE 17th Conf. Digital Signal Processing (DSP)*, Corfu, Greece, July 2011, pp. 1–6.

[25] S. I. Mimilakis, K. Drossos, T. Virtanen, and G. Schuller, “Deep neural networks for dynamic range compression in mastering applications,” in *Audio Engineering Society Convention 140*, Paris, France, May 2016.

[26] J. S. Abel and D. P. Berners, “On peak-detecting and RMS feedback and feedforward compressors,” in *Audio Engineering Society Convention 115*, New York, NY, USA, Oct. 2003.

[27] S. Wei and W. Xu, “FPGA implementation of gain calculation using a polynomial expression for audio signal level dynamic compression,” *Acoustical Science and Technology*, vol. 29, no. 6, pp. 372–377, 2008.

[28] P. Dutilleux, K. Dempwolf, M. Holters, and U. Zölzer, *Nonlinear processing, DAFX: Digital Audio Effects, Second Edition*. Chichester, UK: John Wiley and Sons, Ltd, 2011.

[29] J. Vilkamo, “Alias-free short-time Fourier transform–A robust time-frequency transform for audio processing,” https://github.com/jvilkamo/afSTFT, 2015, [Online; accessed 05-Feb-2017].

[30] J. Breebaart, S. Disch, C. Faller, J. Herre, G. Hotho, K. Kjörling, F. Myburg, M. Neusinger, W. Oomen, H. Purnhagen *et al.*, “MPEG spatial audio coding/MPEG surround: Overview and current status,” in *Audio Engineering Society Convention 119*, New York, NY, USA, Oct. 2005.

[31] D. Giannoulis, M. Massberg, and J. D. Reiss, “Parameter automation in a dynamic range compressor,” *Journal of the Audio Engineering Society*, vol. 61, no. 10, pp. 716–726, Oct. 2013.

A SONIFICATION INTERFACE UNIFYING REAL-TIME AND OFFLINE PROCESSING

Hanns Holger Rutz **Robert Höldrich**

University of Music and Performing Arts Graz
Institute of Electronic Music and Acoustics (IEM)
hanns.rutz | robert.hoeldrich @kug.ac.at

ABSTRACT

SysSon is a sonification platform originally developed in collaboration with climatologists. It contains a domain-specific language for the design of sonification templates, providing abstractions for matrix input data and accessing it in real-time sound synthesis. A shortcoming of the previous version had been the limited breadth of transformations applicable to this matrix data in real-time. We observed that the development of sonification objects often requires pre-processing stages outside the real-time domain, limiting the possibilities of fully integrating models directly into the platform. We designed a new layer for the sonification editor that provides another, semantically similar domain-specific language for offline rendering. Offline and real-time processing are unified through common interfaces and through a mechanism by which the latter can make use of the outputs of the former stage. Auxiliary data calculated in the offline stage is captured by a persisting caching mechanism, avoiding waiting time when running a sonification repeatedly.

1. INTRODUCTION

The *SysSon* sonification platform came out of an eponymous research project running 2012–2015 whose aim was the systematisation of sonification practices [1]. It was designed to specifically handle the type of data produced in climate research, namely multi-dimensional time series, and to be able to translate this data into real-time sound synthesis processes. Another aim was to enable scientists—climatologists in our case—to directly engage with the sonification by running *SysSon* as an application that allows them to import data files and to select, parametrise and listen to sonification models. Development was resumed in 2016 as a case study in knowledge transfer from the artistic into a scientific domain. Working with the same group of climatologists, but using an adapted methodology, we aim at making the platform more practical in typical usage scenarios.

As the design process of sonification requires quick iterations of programming, adjustment and listening, *SysSon* was conceived also as an integrated development environment (IDE) for the sonification design itself, and not just an interface for the deployment of finalised sonification models to scientists as "end users". Consequently, the platform integrates the two perspectives of programming and editing sonification models, on one hand, and the usage and adjustment of sonification models, on the other hand. The former is done through an integrated text editor and compiler—using embedded domain specific language (DSL) extensions for the Scala programming language—the latter through a graphical user interface (GUI).

Until recently, sound computations were restricted to mainly real-time synthesis, based on abstractions provided by a computer music framework [2] that utilises *SuperCollider* in the back-end. For example, there are objects that encapsulate functions that define a *SuperCollider* graph of signal processing blocks or unit generators (UGens). These objects can be copied and moved around in the application and posses a dictionary of parameters, such as scalar values, references to buses, sound files, etc. In the IDE, the UGen graph function is written inside a text editor, using Scala, while the associative dictionaries are generally editable through GUI elements and drag-and-drop.

We have found that the restriction to real-time synthesis can be limiting in many cases, and thus we set out to understand how dedicated pre-processing stages could be added to our system. What we are interested in is to learn how our specific embodied knowledge of the software platform can be analysed through its development process, and how it can be translated to yield a more practical application. The rest of the article is organised as follows: First we analyse how the advantages and limitations of the previous version of the platform led to particular ways of developing sound models. This is elaborated through the specific scenario of sonifying the quasi-biennial oscillation (QBO). We then show how we extended the system by a non-real-time component that in many ways mirrors the real-time API, making it easier to unify the two time domains.

2. DEVELOPMENT PROCESS

It is clear that any decision on design and representations enables particular ways of working with and thinking about the subject matter, as much as they inhibit others. For example, being able to create and program the sound objects from within the application allows them to be conceived as things that can be evolved, duplicated, moved and copied

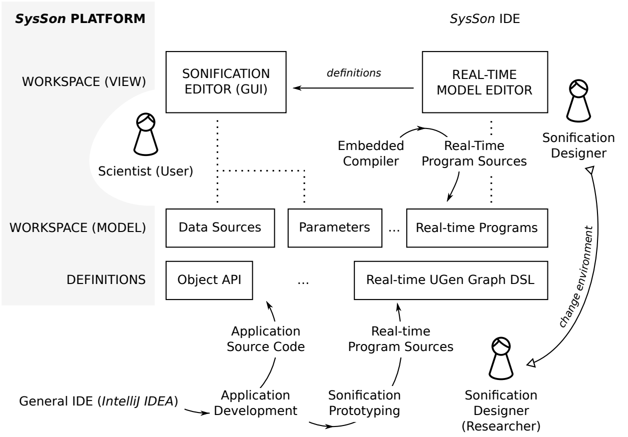


Figure 1. User and developer perspectives and dynamics. This is before the introduction of offline pre-processing.

around between workspaces. The sound objects *live* in workspaces, and in fact the platform can be used to program entire music compositions or sound installations.

On the other hand, our IDE is very rudimentary compared to a professional IDE, when it comes to code navigation, code completion, error checking, break-point debugging and so forth. Also, a monolithic real-time UGen graph function as the embodiment of a sonification process is a strong constraint. Both the delegation of tasks to the graphical interface and the restriction to a single graph function had been perceived as restrictions by some participants of a workshop on *SysSon* we held in 2014. In our own practice of developing sonification models, we found that we could experiment better with some ideas by falling back to a professional IDE (*IntelliJ IDEA*) that was used to develop the application itself, temporarily removing some constraints. This was possible due to the existence of an overall application programming interface (API) extending beyond what is exposed in the GUI. This type of development process, where the sonification designer moves back from the in-application coding to an outside IDE, the same IDE that is used to write the entire application, is illustrated in Figure 1.

2.1 Case Study: Quasi-Biennial Oscillation

To understand this process better, and how it can connect with the in-application work, we present as a case study the creation of a particular sonification model. The scenario in climatology that we were targeting was the quasi-biennial oscillation (QBO) [3]. This phenomenon occurs in the lower stratosphere and equatorial latitudes, and includes temperature anomalies that rhythmically alternate approximately every two years. A temperature anomaly is a deviation of the temperature from its average value at a given time and location. In the QBO, these anomalies oscillate with the temporal phases shifting from higher to lower altitudes, something that can be clearly seen in a plot (Figure 2).

Because of its technical simplicity, many sonification models are built on what has been called *parameter mapping* [4], techniques of more or less directly translating

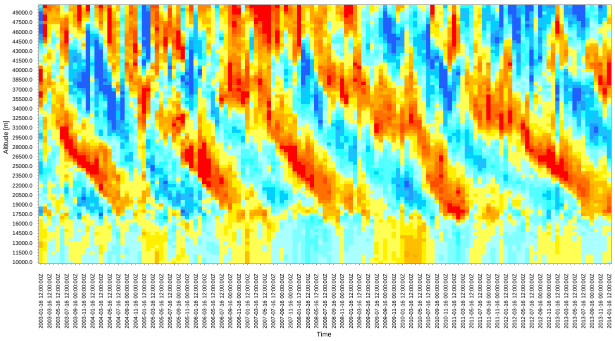


Figure 2. Temperature anomalies ($-5°C$ blue, $+5°C$ red), equatorial in $10km$ to $50km$ altitude, Jan 2003 to Jan 2014.

(mapping) a stream of input data to a stream of data that controls one or several sound parameters. Parameter mapping is also suggested in the design of *SysSon*, as the prominent abstraction added to the set of UGens is `Var`, a reference to matrices and sub-matrices that may then be unrolled as a temporal and/or multi-channel real-time signal that can be treated like any other UGen. While we can create hundreds or even thousands of oscillators or modulators this way without technical problems, obtaining a suitable sonic representation of a multi-dimensional matrix is all but easy. We have tried to address this in the GUI by allowing the user to quickly create sub-selections from matrices and to assign different dimensions to different logical inputs of the sound model.

Nevertheless, it became more and more clear that what was missing from the framework was some general notion of what Grond and Berger call the *data preparation* stage. The data sets in our scenario are stored in *NetCDF* format, and of course it is possible to prepare these sets with various other tools, with numerical computing software, and so forth. The climatologists are used to these tools and perform typical operations such as reducing the resolution of a data set, averaging over a dimension, or calculating the anomalies by subtracting out the monthly averages of a variable. However, we would like to build sonification templates that imply a certain pre-processing without requiring the user to carry it out by hand. This is especially true if we want to explore more unconventional data preparation.

One of the ideas we developed for creating a sonification of the QBO phenomenon is to use an algorithm from the visual domain, a blob detection. Blob detection is an approach from computer vision to determine coherent shapes in a static or moving digital image. The motivation came from the clarity with which we identify the regular diagonal shapes in Figure 2. If we transformed the time-altitude grid of temperature data into a low number of expressive features, we could potentially build a sonification that paralleled the visual clarity in the aural domain.

2.2 Obtaining and Representing Blob Data

To prototype this idea, we looked for a simple open source library that could be used from within the Java Virtual Machine (JVM), since our application runs on the JVM. We

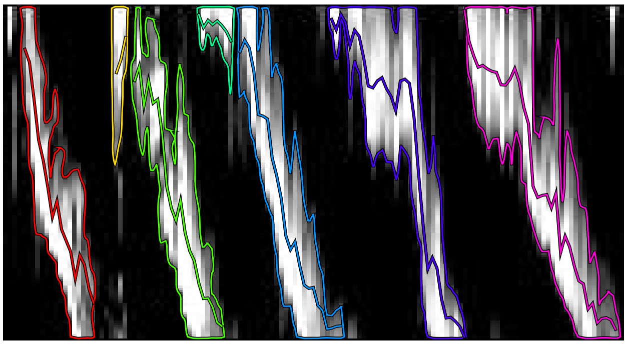
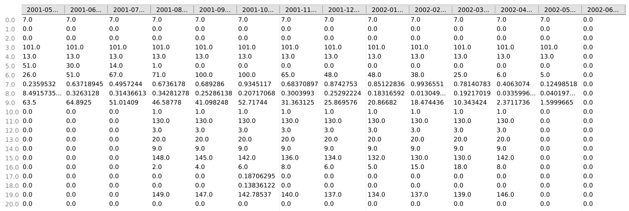


Figure 3. Output of blob detection for positive temperature anomalies (detection threshold $+0.26°C$), equatorial in $21km$ to $36km$ altitude, Jan 2003 to Jan 2014.

found a library written by Julien Gachadoat[1] that produced reasonable results while being implemented in only a few hundred lines of Java code. Within two days, we were able to take a data set of temperature anomalies, convert each latitude/longitude slice into a greyscale image with the time series on the horizontal axis (180 months) and altitudes on the vertical axis (600 levels), run the blob detection, and output a new matrix file containing the evolution of the blob shapes for a given maximum number of concurrent voices. Figure 3 shows the input matrix with blob contours. The polyline inside the blobs is one of the features extracted, the centroid or weighted arithmetic mean, which gives a good representation of the "glissandi" of the anomalies.

How can we make the results of the blob analysis available to the sonification? The solution we picked is to render each blob with one sound generator within a polyphonic UGen graph. The blob data is again rasterised along the time axis, and blobs are pre-sorted into voices. Since there is no support for dynamic growth of voices within a UGen graph, we define a maximum number of concurrent voices at the analysis stage. Theoretically, two voices would suffice to capture the overlapping diagonal shapes, but since the shapes are not fully regular and there occur disturbances to the pattern—visible in the figure by the very short yellow blob, for example—we use four voices *in the prepared data*, and twice as many voices *in the UGen graph*, so that we may employ envelope release phases that extend beyond the strict boundary of one time slice.

If the rasterised and organised blob-data is written out as another matrix file, we thus have a preparation stage that consists of a transformation (input matrices → structured blob-set → equidistant-sampled blob-data → voice distribution → output matrices). The written *NetCDF* matrix file can then be used inside the *SysSon* application like any other sonification data source. A sample section from such a file is shown in Figure 4. The time dimension is preserved and the altitude dimension is replaced by a new dimension that contains the blob "records", each of which occupies ten rows. Without going into the details, each blob record starts with a unique identifier (rows 0, 10, and 20 in the figure) that the sound synthesis will use to trace a coherent object. The identifier is zero whenever the particular voice is not allocated. In the shown table, the top most voice begins at the first time index with blob id 7, a second voice with blob id 1 is allocated in the forth column. Next in the record are the blob dimensions (first altitude level, start time index, number of altitude levels, number of time slices), followed by the features of each time slice, including mean, standard deviation and centroid of the temperature anomalies included (visible in the table by their floating point nature).



	2001-05...	2001-06...	2001-07...	2001-08...	2001-09...	2001-10...	2001-11...	2001-12...	2002-01...	2002-02...	2002-03...	2002-04...	2002-05...	2002-06...
0.0	7.0	7.0	7.0	7.0	7.0	7.0	7.0	7.0	7.0	7.0	7.0	7.0	7.0	0.0
1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3.0	101.0	101.0	101.0	101.0	101.0	101.0	101.0	101.0	101.0	101.0	101.0	101.0	101.0	0.0
4.0	13.0	13.0	13.0	13.0	13.0	13.0	13.0	13.0	13.0	13.0	13.0	13.0	13.0	0.0
5.0	51.0	30.0	14.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
6.0	26.0	51.0	67.0	71.0	100.0	100.0	65.0	48.0	48.0	38.0	25.0	6.0	5.0	0.0
7.0	0.2359532	0.63718945	0.4957244	0.6736178	0.689286	0.9345117	0.68370897	0.8742753	0.85122836	0.9936551	0.78140783	0.4063074	0.12498518	0.0
8.0	8.4915735...	0.3263128	0.31436613	0.34281278	0.25286138	0.20717068	0.3003993	0.25292224	0.18316592	0.013049...	0.19217019	0.0335996...	0.040197...	0.0
9.0	63.5	64.8925	51.01409	46.58778	41.098248	52.71744	31.363125	25.869576	20.86682	18.474436	10.343424	2.3711736	1.5999665	0.0
10.0	0.0	0.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0
11.0	0.0	0.0	0.0	130.0	130.0	130.0	130.0	130.0	130.0	130.0	130.0	130.0	0.0	0.0
12.0	0.0	0.0	0.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	0.0	0.0
13.0	0.0	0.0	0.0	20.0	20.0	20.0	20.0	20.0	20.0	20.0	20.0	20.0	0.0	0.0
14.0	0.0	0.0	0.0	9.0	9.0	9.0	9.0	9.0	9.0	9.0	9.0	9.0	0.0	0.0
15.0	0.0	0.0	0.0	148.0	145.0	142.0	136.0	134.0	132.0	130.0	130.0	142.0	0.0	0.0
16.0	0.0	0.0	0.0	2.0	4.0	6.0	8.0	6.0	5.0	15.0	18.0	8.0	0.0	0.0
17.0	0.0	0.0	0.0	0.0	0.0	0.18706295	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
18.0	0.0	0.0	0.0	0.0	0.0	0.13836122	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
19.0	0.0	0.0	0.0	149.0	147.0	142.78537	140.0	137.0	134.0	137.0	139.0	146.0	0.0	0.0
20.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Figure 4. Section from a rasterised blob matrix

3. INTEGRATED OFFLINE UGEN GRAPHS

The problem with the creation of the blob matrix is, of course, that it is not yet in any way integrated into the application. While this is acceptable for research, we do not want the climatologists to send us their data sets in order to run them through the transformation and send them back to be used in the sonification. A simple solution would be to add an action to the workspace GUI to allow the users to pre-process their matrices and then re-import the resulting blob matrix into the workspace. In fact, we have taken this route before with an action to pre-calculate an anomalies matrix from a regular (e.g. temperature) matrix. But this way, the underlying problem is not addressed: If one can think up and implement a real-time sonification model from scratch with the built-in code editor, why is there no counterpart for the pre-processing stage?

The idea that we have now implemented and that we are now testing is this: **Can we conceive a domain-specific language for offline processing that is relatively easy to understand if one has already learnt how to write the real-time code?** Is the UGen graph model applicable for the pre-processing stage? What are its advantages, disadvantages, impedance matches and mismatches?

The UGen graph concept is well established in sound synthesis and could be seen as a form of dataflow or stream processing model. A network of unit processors is run on a scheduler, each consuming and producing a stream of values. For example, in *SuperCollider*, the UGens form a directed acyclic graph and process floating point values either as chunks at audio sampling-rate, or as individual values at control rate, a fraction of the audio sampling-rate that makes computations less expensive. In the beginning of 2016, we were working precisely on the translation of the UGen graph model to offline processing in the reformulation of the *FScape* audio signal processing software. A few months later, the implementation was stable enough to use it to render not just sounds, but also bitmap images

[1] `http://www.v3ga.net/processing/BlobDetection/index-page-home.html` (accessed 18-May-2017)

```
// real-time
val gen = LFSaw.ar(SampleRate.ir/800, 0.5)
val up  = (gen + 1) * 2
val a   = up.min(1)
val b   = (up - 3).max(0)
a - b

// offline
val gen = LFSaw(1.0/800, 0.75)
val up  = (gen + 1) * 2
val a   = up.min(1)
val b   = (up - 3).max(0)
a - b
```

Figure 5. Comparison of real-time and offline graphs, using the Scala DSLs in *SysSon*.

and video sequences, suggesting that this framework could be also used as the basis for offline processing of matrices in *SysSon*. It is thus useful to summarise the experience with this framework so far:

3.1 Applying the UGen Model to Offline Processing

At first glance, offline and real-time processing should be quite similar, only distinguished by the kind of scheduler that runs the nodes of the graph, in the former case being busy incessantly, and in the latter case paying attention to deadlines. Figure 5 shows the example of an envelope generator in real-time (*SuperCollider*) and in offline (*FScape*) code. The envelope goes from one to zero in 200 frames, then back to one in 200 frames, and remains there for another 400 frames. Three subtle difference can be seen already:

- The phase argument to the sawtooth generator differs. While in *SuperCollider* the ranges of some parameters have historically been chosen to minimise mathematical operations on the server side, in *FScape* we favour consistency across various UGens over additional arithmetical operations required.

- In *SuperCollider* we must specify whether the UGen uses audio or control rate, while in *FScape* these categories do not exist. Instead, the block sizes of signals depend on the particular UGen and may even vary across the inputs and outputs of a UGen, although a nominal block size can be specified that determines the default buffer size for dynamic signals.

- The frequency is given in Hertz in *SuperCollider*. In *FScape* it is given as a normalised value, as there is no notion of a global sampling rate. We were contemplating the introduction of a `SampleRate` UGen for convenience, but since the system was primarily designed to transform audio files, one would mostly want to work directly with the respective sampling rates of the input files. In the cases where images or videos are processed, the notion of audio sampling rate would also be meaningless. In fact, many UGens are now specified to take period parameters in number-of-frames rather than their reciprocal, a normalised frequency, as we also support integer numbers next to floating point numbers, avoiding time-base drift over long runs, something that is known to cause issues in *SuperCollider*.

Property	***SuperCollider***	***FScape***
Language	C, C++	Scala
Scheduler	custom written	*Akka Streams*
Threading	single threaded	optional asynchronous, thread pool
Data rates	dual (audio and control)	multi-rate
Buffering	assigned fixed size "wire-buffers"	variable size, reference passing
Data type	float	unrestricted
Life cycle	all UGens start and stop together	UGens can shut down at different times

Table 1. Comparison of back-end architecture

Both systems use a weak type `GE` (graph element) for the values that can be used as parameters to UGens (constants are graph elements, as are UGens themselves), but while on the *SuperCollider* server all streams are restricted to 32-bit floating point resolution, in *FScape* we adopted multiple numeric types, consisting currently of 32-bit and 64-bit integer numbers and 64-bit floating point numbers. Some UGens are generic in that they can natively work with different numeric types, preserving particular precision and semantics, while others coerce a `GE` into a particular type. For example, `ArithmSeq(0.0, 4)`—similar to `Dseries` in *SuperCollider*—produces the arithmetic series $0.0, 4.0, 8.0, \ldots$ with floating point numbers, while `ArithmSeq(0L, 4)` produces the series $0L, 4L, 8L, \ldots$ with 64-bit integer numbers. This approach is still under evaluation.

The way UGens are constructed from the user-facing side is very similar, including lazy expansion of elements that allows the preservation of pseudo-UGens (e.g. in serialisation) and the late binding of resources needed from the environment (e.g. automatic adaptation of the expanded UGen graph to variable types and dimensions of control inputs), multi-channel expansion, and the composition with unary and binary operators. But when it comes to the implementation of the UGens, the systems use very different architecture, which is summarised in Table 1.

3.2 Interfacing with an Offline Graph

It remains to define an input/output interface for integrating offline processing graphs with other objects inside *SysSon*. All objects share as a common interface a dictionary—called attribute map—that can be used to freely associate attributes with it, using conventional string keys. In the real-time case, for example, when one creates a named control `"freq".kr`, the control input is looked up in the attribute map of a containing `Proc` object (the object wrapping the UGen graph) at key `"freq"`. This could be a scalar value, a break-point function, or any other object that could be

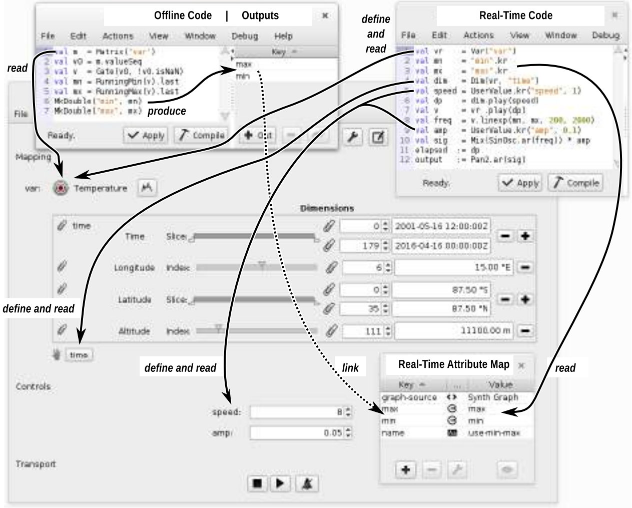
```
val m  = Matrix("var")
val v0 = m.valueSeq
val v  = Gate(v0, !v0.isNaN)
val mn = RunningMin(v).last
val mx = RunningMax(v).last
MkDouble("min", mn)
MkDouble("max", mx)
```

Figure 6. An offline program that determines the range of values of a matrix.

translated into a control signal. The real-time signal output of a `Proc` is represented by a `ScanOut` UGen, for which a dedicated output object must be created inside a special `outputs` dictionary of the `Proc`. This object can then be placed in the attribute map of another object that wants to use the corresponding signal as input. These links thus function as "patch cords" between objects.

In the offline case, we use a similar approach. Here `"freq".attr` would again inject a value or value stream from the containing object's attribute map. Other UGens exists for other types of input: To get access to a matrix, one uses `Matrix("key")` where the matrix is then sought in the attribute map or in the list of variables of the containing `Sonification` object using `"key"`. Likewise, different UGens exists for the different types of outputs produced. Figure 6 gives the example of calculating the minimum and maximum elements of an input matrix. The `valueSeq` method on the matrix produces a one-dimensional "flat" (interleaved) stream of all the elements in the matrix. The `Gate` is used to remove missing values (NaN is for not-a-number), which commonly occur in the climatology data files. The `RunningMin` and `RunningMax` UGens are similar to their *SuperCollider* equivalents, but the `last` operation is unique: In the offline processes, UGens may terminate at different times, once their inputs or outputs are exhausted. The `valueSeq` stream terminates when all elements have been transported, and subsequently `Gate` and `RunningMin/Max` will terminate. The `last` operation produces a UGen that outputs only the last element from its input stream. There are also operations such as `drop` and `take` that act like their counterparts in standard Scala collections.

The last two lines define the outputs of the program. The string parameter defines the key in the output dictionary of the object containing the offline program. Corresponding output objects must be created in that dictionary and may then be linked to the attribute map of a `Proc`, where they are then used in the definition of the real-time process. Figure 7 shows a screenshot to clarify how offline and real-time programs are wired together and integrated within a sonification object. The editor of the offline program is in the top-left, containing the code of Figure 6, along with the two output objects named *min* and *max*. These two objects have been linked through drag-and-drop with the attribute map, shown in the bottom-right, of the real-time object, whose code in turn is shown in the editor window in the top-right.



```
val m  = Matrix("var")
val v0 = m.valueSeq
val v  = Gate(v0, !v0.isNaN)
val mn = RunningMin(v).last
val mx = RunningMax(v).last
MkDouble("min", mn)
MkDouble("max", mx)
```

Outputs: max, min

```
val vr    = Var("var")
val mn    = "min".kr
val mx    = "max".kr
val dim   = Dim(vr, "time")
val speed = UserValue.kr("speed", 1)
val dp    = dim.play(speed)
val v     = vr .play(dp)
val freq  = v.linexp(mn, mx, 200, 2000)
val amp   = UserValue.kr("amp", 0.1)
val sig   = Mix(SinOsc.ar(freq)) * amp
elapsed  := dp
output   := Pan2.ar(sig)
```

var: Temperature

time				
Time	Slice	0	2001-05-16 12:00:00Z	
		179	2016-04-16 00:00:00Z	
Longitude	Index	6	15.00 °E	
Latitude	Slice	0	87.50 °S	
		35	87.50 °N	
Altitude	Index	111	11100.00 m	

time

speed: 8

amp: 0.05

Key	Value
graph-source	Synth Graph
max	max
min	min
name	use-min-max

Figure 7. Interlinkage of offline and real-time programs within the application's user interface

The large window in the background shows the sonification editor GUI as it is presented to the user, e.g. the climatologist. The interface elements have been programmatically generated by the real-time program (this is still considered the main entry to the sonification model, while the offline programs function as auxiliary components): `Var` refers to an input variable like `Matrix`, but additionally defines a corresponding user interface element in the sonification editor. `Dim` UGens allow the user to associate dimensions of the sonification model with dimensions of the matrices sonified. The `UserValue` UGen, used for speed and amplitude here, is a control like `"key".kr`, but also requests the presentation of a user interface element. In a future version, we envision a set of dedicated user interface objects that can be linked to standard controls, disentangling the real-time program from decisions on the user interface.

3.3 Data Preparation Stage

We now return to the blob example. In order to reformulate the data preparation, we need to create new UGens that encapsulate the blob detection and analysis. Generally, the best approach is to minimise the functionality of a UGen, perhaps splitting an algorithm across a number of UGens, so that partial functions can be reused in other contexts. An obvious choice would be one UGen `Blobs2D` for the blob detection stage, yielding the contours of the blobs within each matrix slice, and a second UGen `BlobVoices` that transforms this set of geometric objects into the rasterised matrix of voices containing the required analysis of the blob contents, such as the sliding extent and centroid. The program complete with input UGen `Matrix` and output UGen `MkMatrix` is shown in Figure 8.

Without going into much detail, we want to highlight three aspects of this reformulated algorithm:

Program storage: We store a program as a tree of (unexpanded) UGens and not the source code or the compiled

```
val voices  = 4    // maximum polyphony
val taLo    = 0.0  // lowest  temp anomaly considered
val taHi    = 3.5  // highest temp anomaly considered
val thresh  = 0.26 // blob detection threshold

val mIn     = Matrix("anom")
val d1      = Dim(mIn, "time")
val d2      = Dim(mIn, "altitude")
val width   = d2.size
val height  = d1.size
val blobSz  = voices * 10 // blob record has 10 elems

// transform the matrix spec (shape)
val specIn  = mIn.spec
val s1      = specIn.moveLast(d1)
val s2      = s1    .drop    (d2)
val d3      = Dim.Def("blobs", values =
    ArithmSeq(length = blobSz))
val specOut = s2    .append  (d3)

val win0    = mIn.valueWindow(d1, d2)
val win1    = Gate(win0, !win0.isNaN)
val win     = win1.max(taLo).min(taHi) / taHi

val blobs   = Blobs2D(in = win, width = width,
    height = height, thresh = thresh, pad = 1)

val winB    = BufferDisk(win)
val mOut    = BlobVoices(in = winB, width = width,
    height = height, numBlobs = blobs.numBlobs,
    bounds = blobs.bounds, numVertices =
    blobs.numVertices, vertices = blobs.vertices,
    minWidth = 10, minHeight = 4, voices = voices)

MkMatrix("out", specOut, mOut)
```

Figure 8. UGen-based formulation of the blob matrix preparation

bytecode. This has two reasons: First, Scala is a statically typed and compiled language, and while we embed an on-the-fly compiler for read-eval-print-loop (REPL) functionality, the compilation is slow and resource hungry. In the real-time case, compiling on-the-fly is prohibitive, and regular language control structures such as `if-then-else` blocks cannot be represented on the sound synthesis server, so there is little benefit to store Scala programs. For the offline case, storing regular (compiled) Scala programs is still an interesting option, and something that is done already elsewhere in the application for `Action` objects. The problem here is compatibility—on the one hand, Scala has adopted a policy of allowing binary incompatibility between major release versions in order to facilitate language evolution. *SysSon* has recently been migrated from Scala 2.11 to 2.12, so if we had stored binary code programs, we would face the problem of automatically detecting incompatible versions when opening an old workspace and recompiling them. An interesting new possibility will arrive in the future when Scala has migrated to a new serialisation format called *TASTY* [5] that promises to address this. But even then, a danger remains that closures may unintentionally capture variables in the system (cf. [6]). The second reason is that we would have to give much stronger guarantees about the stability of the entire API that is exposed if we stored the binary program. Storing the UGen graph has been proven a good solution, because it still very expressive as it basically corresponds to the abstract syntax tree (AST) of the DSL, and we can change implementation details without rendering the programs incompatible.

The isolation of the DSL has brought another tremendous advantage, as we can compare two programs for structural equality. This is crucial to correctly **cache the results** of the offline program. Because analyses like the blob detection can take some time, being able to cache the results allows the user to quickly adjust sonification parameters without being penalised by long waiting times, as long as the parameters do not affect the structural equality of the program (e.g. a different part of the input matrix must be analysed).

Manipulating structural data: Partly related, a recurring problem or question is the transport of structural data that is not well captured by the streaming of plain numeric values. This is clearly showing here in the adaptation of the matrix specification from input to output: We need to be able to remove the altitude dimension and replace it by a new "record" type of dimension for the blobs here, the transformation from `specIn` to `specOut`. The DSL must foresee all possible scenarios since, because of the program storage constraints, it is not possible to use the expressive Scala language feature of applying a higher-order function, e.g. writing `specIn.filterNot(_.name == d2.name)`. But even if this was possible, we have a mismatch with the streaming protocol. In `Blobs2D`, we have regular vectorial data of fixed window sizes flowing into its input, but for each window we have a varying (much smaller) number of hierarchical output records that would ideally be objects in the form of

```
class Blob(bounds: Rectangle, slices: List[Slice])
class Slice(extent: Int, mean: Double, std: Double)
```

Remarkably, there are only few discussions on representing record-type data in streaming systems. There was work on this for *FAUST* [7], but even there records are supposed to bundle together flat objects of equal rate components, and nested structures are not addressed. Systems either force all data into vectorial representations—for instance, *OpenCV* puts polygon output from analysis into a standard matrix, just as we output the blob contours as a stream of pairs of x/y coordinates demarcated by an independent outlet for `numVertices`—or they divide the world into vector data and message or event data, as in *Pure Data* and *Max/MSP*, where records can be represented as message lists, although the problem of nested data structures is addressed just as little.

Multi-rate alignment: The line `val winB = BufferDisk(win)` indicates a problem. The streaming infrastructure we have chosen, *Akka Streams*, uses back-pressure, a combination of dynamic push (data-driven) and pull (demand-driven) approaches with bounded queues between nodes. This is generally helpful when real-time properties are not required, because the system tends to adjust itself to the particular data-rates produced in the nodes, while automatically blocking unlimited growth of unprocessed messages (resulting in running out of memory), where a consumer node cannot catch up with the pace of a producer node. Nonetheless, whenever the program contains a diamond topology—a sink and source are connected by more than one path—this can result in a stalling process, and appropriate intermediate buffers must be inserted. *Akka* currently has no means to automatically detect these situations, and so it is the programmer's responsibility to manually insert buffers. `BufferDisk` is the most simple form of such a buffer, as it is unbounded and written to disk if it becomes too large. Alternatively, we could have just duplicated the `valueWindow` call, although we would then also duplicate a few more nodes and thus require more computation.

In a traditional signal processing scenario, it is easy enough to calculate the necessary buffers, and a future version of *FScape* can probably accomplish this automatically, but with non-vectorial structural data such as the output of the blob detection, we fear that manual control will still be required. Again, some research on multi-rate management exists, e.g. [8], but this is focused on real-time scenarios (with properties such as de-coupling that do not apply here) and vectorial data. Another related aspect is the complexity of implementing multi-rate UGens with many inputs and outputs. As Arumí Albó has noted, handling multi-rate signals inside modules "yields complex code in every concrete module" [9, p. 145].

4. DISCUSSION AND CONCLUSION

We have presented a new architecture within the *SysSon* sonification platform to integrate real-time sound synthesis with offline pre-processing stages, using a similarly constructed DSL for writing UGen graphs. The work is the result of quick iterations that responded to feedback from climatologists who we are trying to give a tool at hand that they can use all the way from data import to model selection and parametrisation. The architecture was elaborated through the case study of using a blob detection algorithm to sonify the QBO phenomenon. After presenting a set of rendered sounds to the climatologists, [2] it became quite clear that allowing them to navigate interactively through the data and adjust different sound parameters would be a requisite for advancing the sound models, so having an integrated data pre-processing stage was an important step towards this goal.

Another important result of this research is a better understanding of the development and transfer processes that take place at the various stages of experimenting with new sonification approaches and feeding them back into a stable platform. It emerges a pattern of oscillation in this process between the greenfield experimentation in the external IDE, and reformulation within the inner IDE for deployment. This movement is illustrated in Figure 9.

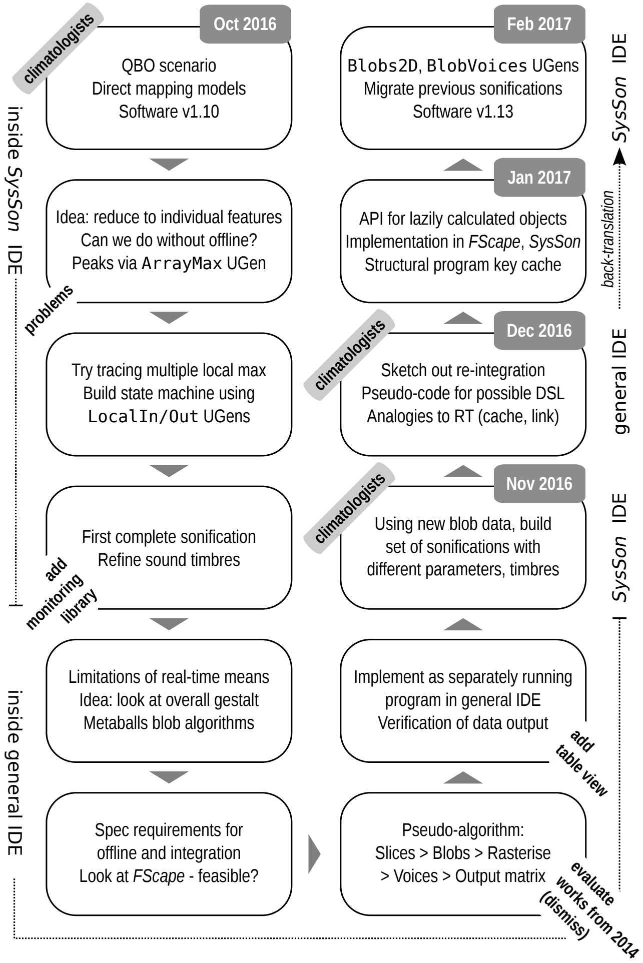

climatologists
Oct 2016
QBO scenario
Direct mapping models
Software v1.10
Feb 2017
Blobs2D, BlobVoices UGens
Migrate previous sonifications
Software v1.13
SysSon IDE
inside SysSon IDE
Idea: reduce to individual features
Can we do without offline?
Peaks via ArrayMax UGen
Jan 2017
API for lazily calculated objects
Implementation in FScape, SysSon
Structural program key cache
back-translation
problems
Try tracing multiple local max
Build state machine using
LocalIn/Out UGens
climatologists
Dec 2016
Sketch out re-integration
Pseudo-code for possible DSL
Analogies to RT (cache, link)
general IDE
First complete sonification
Refine sound timbres
add monitoring library
climatologists
Nov 2016
Using new blob data, build
set of sonifications with
different parameters, timbres
SysSon IDE
inside general IDE
Limitations of real-time means
Idea: look at overall gestalt
Metaballs blob algorithms
Implement as separately running
program in general IDE
Verification of data output
add table view
Spec requirements for
offline and integration
Look at FScape - feasible?
Pseudo-algorithm:
Slices > Blobs > Rasterise
> Voices > Output matrix
evaluate works from 2014 (dismiss)

Figure 9. Stages in the development of the blob sonification

A number of concrete suggestions can be made with respect to the future development of this architecture:

- It would be extremely interesting to have an interactive worksheet or REPL mode for offline programs. That is to say, the ability to execute these programs step by step, validate the intermediate outputs, readjust, etc., akin to the "notebook" mode of operation that has gained much traction with *iPython* and *Jupyter* [10].
- Offline programs could become a formalisation within the application code base itself for various situations:
  - Real-time UGens could be added that implicitly spawn pre-processing stages, freeing the authors of sound models to create separate offline objects for trivial tasks. For example, querying the minimum and maximum value of matrix would be better suited as operators directly available within the real-time program.
  - The user (scientist) facing side of the editor could have a richer palette of matrix transformation operators. For example, ad-hoc averaging across a dimension or down-sampling would be a very useful operation offered to the user, and of which a particular sonification model

[2] See the revised sets labelled `WegC_170508-...` at `https://soundcloud.com/syssonproject/sets/` (accessed 18-May-2017)

would not need to have any knowledge or presumption. Such additional matrix reduction or transformation operations could transparently spawn corresponding offline programs before feeding the sonification.

  - The program already provides a set of dedicated transformation utilities, e.g. for the calculation of anomalies. These could now be implemented coherently through the offline DSL.

- The UGen implementation is quite involved, especially when structural and multi-rate data is needed. We suggest to define another intermediate layer on top of *Akka Streams* that provides more formal definitions for finite state machines. The ad-hoc state machine for `BlobVoices` is several hundred lines of code, and thus way too complex. We need better abstractions or macros for the implementing classes. Had we fused `Blobs2D` and `BlobVoices` into the same UGen, the process would have been obviously much simpler—as we would have skipped the stream representation of nested data structures—but this experiment was exactly aimed at seeing if passing around structural data was feasible within the existing streaming framework. We can conclude that it is feasible but demands better library support to make writing these types of UGens less work and error-prone.

An overall picture emerges, where we need to reconsider the perspective on sonification. The existing system makes strong presumptions about the "object" nature of sonification—we have *a sonification*, we apply *a sonification*, a sonification *is a* sound process coupled with an associative dictionary of data sources, etc. Paying more attention to the details of the development process can help to understand sonification rather as a process itself, which brings into focus the translational competency of artistic researchers working in the field. Consequently, the challenge will be how this process view can be married to the (rightful) expectation of scientists to adopt these systems as stable and clearly delineated tools.

Acknowledgments

The *SysSon* platform was originally developed within project P 24159, funded by the Austrian Science Fund (FWF). Since 2016, the continued development at the Institute of Electronic Music and Acoustics (IEM) of the University of Music and Performing Arts Graz is funded by the Knowledge Transfer Centre South (WTZ Süd) Austria. Climate data sets and evaluation feedback is kindly provided by the Wegener Center for Climate and Global Change (WegC) at the University of Graz.

References

[1] K. Vogt, V. Goudarzi and R. Höldrich, 'SysSon – a systematic procedure to develop sonifications', in *Proceedings of the 18th International Conference on Auditory Display (ICAD)*, Atlanta, GA, 2012, pp. 229–230.

[2] H. H. Rutz, K. Vogt and R. Höldrich, 'The SysSon platform: A computer music perspective of sonification', in *Proceedings of the 21st International Conference on Auditory Display (ICAD)*, Graz, 2015, pp. 188–192. [Online]. Available: `http://hdl.handle.net/1853/54126`.

[3] M. P. Baldwin, L. J. Gray, T. J. Dunkerton, K. Hamilton, P. H. Haynes, W. J. Randel, J. R. Holton, M. J. Alexander, I. Hirota, T. Horinouchi, D. B. A. Jones, J. S. Kinnersley, C. Marquardt, K. Sato and M. Takahashi, 'The quasi-biennial oscillation', *Reviews of Geophysics*, vol. 39, no. 2, pp. 179–229, 2001. DOI: `10.1029/1999RG000073`.

[4] F. Grond and J. Berger, 'Parameter mapping sonification', in *The Sonification Handbook*, T. Hermann, A. Hunt and J. G. Neuhoff, Eds., Berlin: Logos Verlag, 2011, pp. 363–397.

[5] M. Odersky, E. Burmako and D. Petrashko, 'A TASTY Alternative', EPFL, Lausanne, Tech. Rep. 226194, 2016. [Online]. Available: `https://infoscience.epfl.ch/record/226194`.

[6] H. Miller, P. Haller and M. Odersky, 'Spores: A type-based foundation for closures in the age of concurrency and distribution', in *Proceedings of the 28th European Conference on Object-Oriented Programming (ECOOP)*, Uppsala, 2014, pp. 308–333. DOI: `10.1007/978-3-662-44202-9_13`.

[7] P. Jouvelot and Y. Orlarey, 'Dependent vector types for data structuring in multirate faust', *Computer Languages, Systems & Structures*, vol. 37, no. 3, pp. 113–131, 2011. DOI: `10.1016/j.cl.2011.03.001`.

[8] G. Essl, 'Playing with time: Manipulation of time and rate in a multi-rate signal processing pipeline', in *Proceedings of the 38th International Computer Music Conference*, Ljubljana, 2012, pp. 76–83. [Online]. Available: `http://hdl.handle.net/2027/spo.bbp2372.2012.013`.

[9] P. Arumí Albó, 'Real-time multimedia computing on off-the-shelf operating systems: From timeliness dataflow models to pattern languages', PhD thesis, Universitat Pompeu Fabra, Barcelona, 2009.

[10] H. Shen, 'Interactive notebooks: Sharing the code', *Nature*, vol. 515, no. 7525, p. 151, 2014. DOI: `10.1038/515151a`.

QUALIA: A Software for Guided Meta-Improvisation Performance

Manuel Brásio
University of Porto,
Faculty of Engineering,
- Porto, Portugal
mbrasio@sapo.pt

Filipe Lopes
University of Aveiro
- Aveiro, Portugal
filipelopes@ua.net

Gilberto Bernardes
INESC TEC
- Porto, Portugal
gba@inesctec.pt

Rui Penha
INESC TEC and
University of Porto,
Faculty of Engineering,
- Porto, Portugal
rui.penha@inesctec.pt

ABSTRACT

In this paper we present *Qualia*, a software for real-time generation of graphical scores driven by the audio analysis of the performance of a group of musicians. With *Qualia*, the composer analyses and maps the flux of *data* to specific score instructions, thus, becoming part of the performance itself.

Qualia is intended for collaborative performances. In this context, the creative process to compose music not only challenges musicians to improvise collaboratively through active listening, as typical, but also requires them to interpret graphical instructions provided by *Qualia*. The performance is then an interactive process based on "feedback" between the sound produced by the musicians, the flow of data managed by the composer and the corresponding graphical output interpreted by each musician. *Qualia* supports the exploration of relationships between composition and performance, promoting engagement strategies in which musicians participate actively using their instrument.

1 INTRODUCTION

Traditional Western music notation privileges pitch and duration over timbral attributes (which apart from instrumentation) are hardly considered [12]. It's longstanding use suggests its effectiveness, however, recent compositional approaches have been providing alternate notation strategies to enhance dimensions of sound (notably timbre), which have been neglected so far. Among these, we can highlight compositions exploring graphical scores as a medium for music notation and expression. Graphical scores represent one approach to handle such aspects but, at the same time, challenge musicians in aspects such as: how to interpret, how to map graphics to sound or how to rehearse such scores. Some examples of graphical scores are included in Cage's book *Notations*, Sauer's *Notations21* or Villa Rojo *Notación y grafía musical en el siglo XX*. This book, besides the graphical scores itself, reflect composer's concerns related to the relationship between musicians, interpretation, and graphical representation of sound and sound itself [13]. Nonetheless, these are graphical scores that are "frozen" just like conventional notation, which means that such scores are static. Musical works like *Cypher* by Pedro Rebelo, *Six Graphic Scores* by John Teske, *Constelazzione* by Lombardi or *4 Systems* by Earle Brown, represent such approach.

Composers like Ryan Ross Smith, Cat Hope, Candas Sisman, Jesper Pedersen, Lindsay Vickery or Leafcutter John have also used graphical scoring. Their work is focused on developing animated notations (i.e. not static) and developing graphics that can be easily grasped by performers [10]. About this subject, Ryan Ross Smith tells us that animated notation "does not lead to a random performance or purely free improvisation. The composer defines the limits. Animated notation is simply a different approach to music composition and interpretation."[10].

In the last decade, there has been a growing interest in generating scores in real time whether graphical, traditional or mixed. Mapping *data* to generate scores in real time has been one of the main challenges. [3] While some approaches rely on pre-defined mappings [3], others require an operator (e.g., composer) to manipulate and map the data (e.g. *Õdaiko* by Filipe Lopes, *Peripatoi* by Rui Penha, *Spam* by Luciano Azzigotti or *Leave no Trace* by Michael Alcorn). Mixed aproches can be found in (e.g. *Zero Waste* by Didkovsky, *Anticipatory Score* by Wyse and Yew). This trend has been capturing the attention of the academic community, especially since computers are able to generate graphics in real-time and communicates wirelessly. [1]

Qualia collects inspiration from all these approaches. The issue that formed the base of its development is: How to manage the mapping of audio *data* to guide a performance using graphical scores generated in real time?

2 QUALIA

Qualia is a digital application that generates graphical music scores in real-time. It targets collaborative performances, in which the audio features extracted from a live instrumental performance drives the generation of graphics. A computer operator is then

[1] The International Conference on Technologies for Music Notation and Representation (TENOR), born in 2015, is dedicated to this area of investigation.

responsible for routing the graphics resulting from the input signals analysis to the instrumentalists, thus creating a feedback system with complex mapping and contrapuntal capabilities. The system was developed in *Max/MSP* [4] and *Processing* [10]. *MaxMSP* is used to analyse and process the live audio input signals, which result is then sent to *Processing*, that interprets the data to create graphics on a rolling screen.

2.1 Flow of data

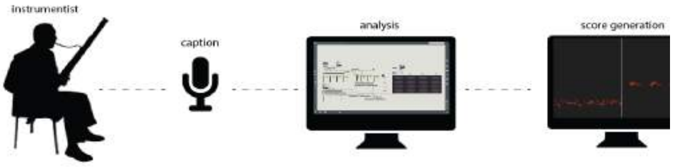


Figure 1. Example of an Individual Analysis and Score Generation scheme
(instrumentist / caption / analysis / score generation)

The input of the system is based on the sound collection. In this case is produced by musical instruments. The audio stream of each instrument is captured individually. For each stream a Bark spectrum representation is extracted using the ZSA.Descriptors library [7]. In greater detail, the audio input signal is represented by a list of 24 values, which express the loudness of logarithmic-distributed frequency bands closely aligned to the psychophysiological parsing mechanics of the auditory human system. [14] Finally, the Bark spectrum analysis of each instrument is sent to Processing using the Open Sound Control (OSC) protocol. Fig. 1 shows the an Individual Analysis and Score Generation scheme, Fig. 2 presents one of the musicians playing with an visual input and Fig. 6 shows a rehearsal moment.

This work was conceived as a musical creation tool in performance. This was idealized from the beginning to be presented in concert in order to validate its usability. For that, a quartet of different instruments was chosen: electric guitar, bassoon, cello and percussion. This diversity of instruments made it possible to perceive the diversity of reactions that the system would generate to different inputs. Each player has its particular computer monitor in front, which displays the graphics as a rolling score that represents a window of about 7 seconds, on the individual screen, and 15 seconds on the group screen.

Figure 2. Cellist experiencing the suggestion monitor side of *Qualia*.

The screen is divided into two parts: a) from the centre to the left, it shows the representation of the sound analysis of the instrumentalist in real time; b) from the right border to the central bar, the gesture indication suggested by the pilot composer. The drawn graphics include circumferences expressing the loudness of each Bark frequency band at each 1024 ms analysis slice (represented vertically). To distinguish participating instruments different colours are applied. Fig. 3 shows an individual monitor.

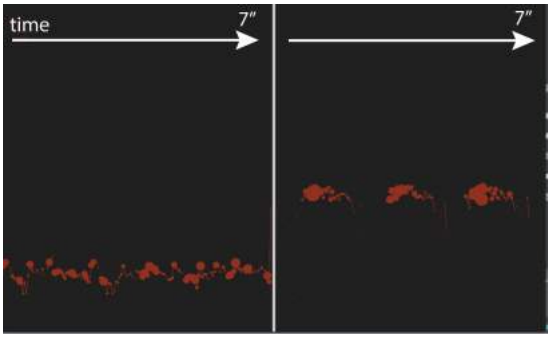


Figure 3. Individual Score.
[a) – Real Time Representation / b) – Graphical Suggestion]

The composer / pilot, as well as the audience during the performance, has access to the overall visualization that is being created according to the same process of the previous one - figure 4 - with the exception of only presenting the final real-time result of the four instruments**.**

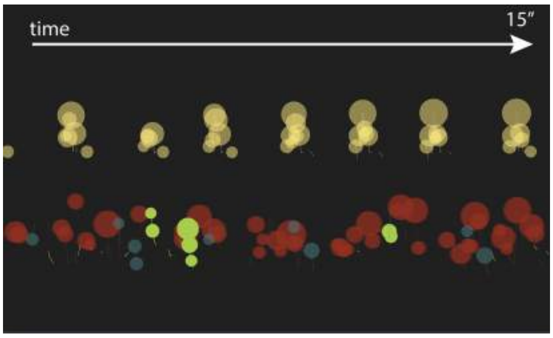


Figure 4. Full Score.

The composer / pilot interface consists of two MIDI controllers that command the signal input level of the Max sliders and simultaneously activate the matrix buttons that direct the signal between the various players' screens. The returning of the information to the player is achieved through a matrix in which the pilot/composer redirects that information through buttons that control the matrix and sliders that control the amount of signal that is sent.

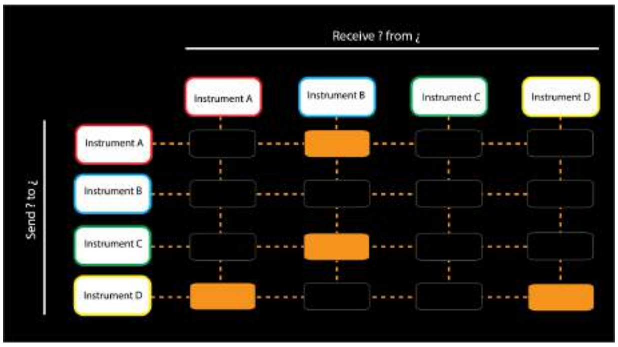


Figure 5. Instrument Matrix

Figure 5 shows a diagram of the matrix that controls the signal flow between the different players:

- The player A is receiving information from the player D.
- The player B simultaneously receives information from the player A and C.
- The player C is in free mode – he is not receiving any suggestions beyond the representation of him or herself (left side).
- The player D receives his own information on both sides of the screen.

3 COMPOSING WITH QUALIA

Creating music with *Qualia* required some time of experimentation, which naturally developed in three tests stages to develop a satisfactory working method.

The first stage was an individual test with the musicians. Each player was submitted to a set of software rehearsal tests, which aimed to analyse how the player related with the graph. In order to understand what would work best, were used some different graphics until was reached an effective version. Each musician was asked to freely experiment the reaction of the graph to what was presented to him or her and then to respond to a list of gestures and instructions given by the composer.

The second stage was combined a quartet of musicians in a similar set of tests. We analysed the different reactions between the graphs that travelled from one to another. We played the *telephone game*[2] with basic gestures like "*short attack*" or "*descending melodic line*" - in this version of the game the message was controlled by the composer and passed through the screen of each musician. During the group tests, some questions were raised by spontaneously discussing the reading and interpretation possibilities of what was exposed to them. Each player realized in what ways he/she could reorganize the ideas presented, in what aspects he could have to adapt his instrument to the approaching chart and, above all, to the music that was being created.

The last stage was the most exciting and challenging. Until this moment, the musicians had acquired some notions of how to interpret the graph they saw, which was a representation of a sound that they heard a few moments before. After several approaches to reading techniques were compared, the exchange of ideas and the rules of reading and interpreting messages were tested, we had to find some ways of overcoming the literal view of a graph and interpreting it in other ways, making music, as well as some rules of interpretation to support the performance. The answer to these questions came naturally as a set of rules was established so as to help to construct a balanced performance as shown in fig.6:

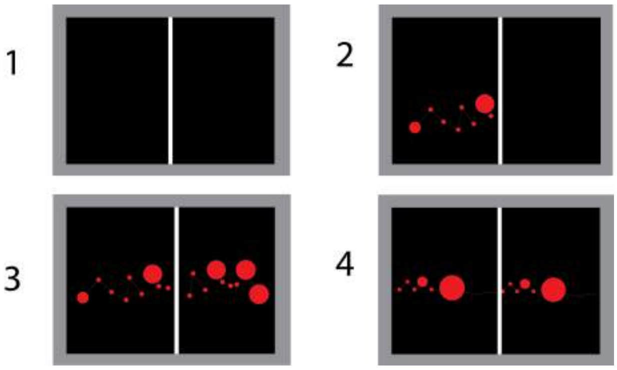


Figure 6. Instrument Matrix

1 - If both parts of the screen are off, do not play.
2 - If you have only direct return of your intervention (left side) you can play freely.
3 - If you have suggested musical information (right side) you should try to adapt your interpretation to the present musical moment.
4 - If you have both sides reacting equally to your interventions, you must assume that you are in solo mode and are free to significantly change the character of the joint musical result.

From this moment forward, we had only to rehearse these ideas. It was possible to perceive what we suspected, the more contact the musicians had with the graphic, the more comfortable they felt, and this was audible in the quality of the music they made. Through several conversations with them, it was possible to perceive that the graphic had stopped to be a merely imposing score of a musical order to accomplish. It was now a hint to a direction they could give to their performance. All the rest was communication in the form of sounds and graphics.

When this system is referred to as an Guided Software for Meta-Improvisation, it is intended to make it clear that *Qualia* is an interactive musical system managed by a composer / pilot, where each musician is constrained by a set of stimuli, graphics and sound generated in real time by other players, which influences him or her in their freedom of improvised performative decision.

[2] Telephone Game is an internationally popular game in which one person whispers a message to the ear of the next person through a line of people until the last player announces the message to the entire group

4 CONCLUSIONS AND FUTURE WORK

The main premise of this project was the production of a multimedia tool capable of creating improvised performances, through real-time composition, based on ideas triggered by the musicians that are symbolized by a representation graph of their sounds. It was while searching for a unique tool that it was concluded that the best option for this project would be a mixture of the advantages of all these concepts.

After all the tests, trials, final concert and the analysis of the results, it is safe to say that *Qualia* proved to be an effective system for the guidance of improvised performative motions, as intended; The musicians have connected to a type of graphic that was introduced to them as an interactive representation of the sound they produced, thus allowing for the communication and performative control of a small ensemble through graphics arising from this analysis, which are shared among the various participants. Concerning the first reactions to the graphic, the musicians revealed some strangeness and confusion towards a graphic that was drawn from impulses created by them. It took some experiments for a few weeks to make the relationship closer and to increase the intuitive reaction to the stimulus. We can say that this system is effective for composer-interpreter communication. While not creating an immediate connection, musicians feel tempted to experiment further and quickly create a natural connection to the stimulus. *Qualia* also enables the technical development and the intuitive approach of the musicians to their instrument and, consequently, to the perception of the possibilities of their surroundings and communication with others. And as a result it was possible to create music that is not only based on the mixing of gestural habits and intrinsic to player, but results from a sharing of experiences and knowledge.

As a summary, it may be noted that the system surprised the musicians. They had never been contextualized in this type of creative practice using multimedia tools. On the other hand, as the project advanced and the experience within the system increased, it was also from that freedom that they had greater pleasure from the alternatives of reading to a visual stimulus and the counterpoint of another visual impulse.

Figure 7. Rehearsal Moment.

4.1 Work Links

All this work was started in Manuel Brásio Masters Thesis in Multimedia: Interactive Music and Sound Design from Faculty of Engineering, University of Porto. All the interested ones can access the following link to download the document in Portuguese: https://sigarra.up.pt/feup/pt/pub_geral.show_file?pi_gdoc_id=792261, and this next link to the website of the main author where it is possible to find more audio-visual details about the project: http://manuelbrasio.xyz/research

4.2 Future Work:

In the near future, we will seek to implement this system on several operating systems and look for a solution that allows the reduction of the number of machines required. We also intend to further explore these ideas and to understand if intervention in the perfomative and creative role of the musicians is indeed enriching. We also think that Qualia could have a lot of potential in the pedagogical environment. Where it can be used as a tool for gestural exploration, improvisation and sound awareness of young musicians.

Qualia showed to be the concretization, even if it is no more than a prototype, of a compositional and performative idea, with a strong intervention function in the role of the musicians in the music they make, and with the potential to be deepened.

Acknowledgements

We would like to thank to Faculty of Engineering, University of Porto and to all the others involved in the different stages of this project.

This project was partially supported by FCT, the Portuguese Foundation for Science and Technology, under the post-doctoral grant SFRH/BPD/109457/2015.

5 REFERENCES

1. Alcorn, Michael, Christopher McClelland. “Exploring new composer/performer Interactions using Real-Time Notation” Sonic Arts Research Centre Queens University Belfast, 2008
2. Brásio, Manuel. “Qualia – Aplicação Digital de Orientação para Meta-Improvisação” – Faculdade de Engenharia da Universidade do Porto, 2016
3. Barrett; G. Douglas, Michael Winter and Harris Wulfson “Automatic Notation Generators”, State University of New York at Buffalo; University of California, Santa Barbara and Graduate Center of City University of New York, 2007
4. MaxMSP - https://cycling74.com/

5. Fisher, Christian M. “Undestanding Animated Notation” Estonian Academy of Music and Theater, 2015

6. Guedes, Carlos. “A Lição sobre Composição em Tempo Real”, Escola Superior de Música e Artes do Espectáculo do Porto, 2008

7. Hope, Dr Cat and Lindsay Vickery. “Screen Scores: New Media Music Manuscripts.” 2011

8. Jordan, Emmanuel and Mikhail Malt. “Zsa.Descriptors: a library for real-time descriptors analysis”.http://www.e--j.com/index.php/download-zsa/ , 2009

9. Lopes, Filipe. “Õdaiko: A real time score generator based on rhythm”. Escola Superior de Música e Artes do Espectáculo do Porto, 2009

10. Processing - https://processing.org/

11. Smith, Ryan Ross. “An Atomic Approach to Animated Music Notation” Rensselaer Polytechnic Institute, 2015

12. Wishart, Trevor, “On Sonic Art”, Harwood Academic Publishers,1996

13. Wyse, Lonce and Jude Yew. “A Real-Time Score for Collaborative Just-in-Time Composition”, 2014

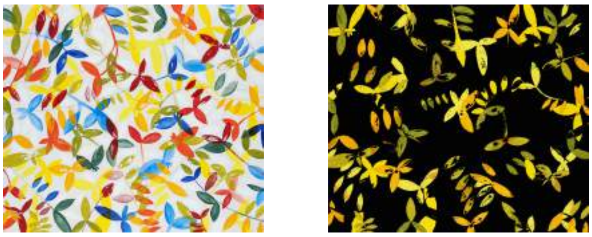
14. Zwicker, Eberhard. “Subdivision of the audible frequency range into critical bands (Frequenzgruppen),” Journal of the Acoustic Society of America,1961

INTERACTIVE PAINTING SONIFICATION USING A SENSOR-EQUIPPED RUNWAY

Edoardo Micheloni, Marcella Mandanici, Antonio Rodà, Sergio Canazza
Dept. of Information Engineering
University of Padova (Italy)
{micheloni, mandanici, roda, canazza}@dei.unipd.it

ABSTRACT

In the present study we will show the realization of a multimedia installation for the interactive sonification of a painting of Hartwig Thaler. The sonification is achieved by means of sound synthesis models, capable of reproducing continuous auditory feedback, eventually generated from the feature analysis of three color layers extracted from the painting (red, blue, yellow). Furthermore, a wooden sensors equipped runway, divided in three sections, each of which represent a different layer, will activate the corresponding soundscape according to the visitor position. This system enables to discover a 2D painting in a new 3D approach, moving along the runway toward the projection of the painting. The paper describes in details the developing of all the elements of the system: starting from the analysis of the image and the algorithm to realize the acoustic elements, than the sensorized runway and finally the testing results.



(a) The original image (b) The yellow matrix

Figure 1: Example of image processing from the original painting to a single color mask.

1. INTRODUCTION

The conjunction of real-time computer graphics, sensors and synthesized music is creating a new medium where people are physically immersed in an environment by a world of images, sounds, and objects that react to their actions. In the late '60s and '70s, installations became a favorite form for the artists working against the notion of the permanent, and therefore collectable, art object. It is from the union of interactive spaces and multimedia installation that the idea of the present work took shape. Interactive installations are a sub-category of art installations and, for instance, can be electronic-based, mobile-based, web-based, etc [1]. They frequently involve the audience acting on the work of art, increase their involvement [2, 3]. Several experiments have been carried out on the augmentation art museums with interactive technology. They may include, for instance, the project [4] that introduce the notion of 3D sound in headphones for an art museum, providing the user with a contextual and spatial audio guide. In [5], the idea is to use computer graphics and augmented reality techniques in order to provide projected overlays on backgrounds with arbitrary color and reflectance. Another project may refer to is [6, 7] where an ancient musical instrument can be virtually played thanks to sensors and a touch interfaces. One of the most representative exponent of virtual reality is surely Myron Krueger. He is considered to be one of the first-generation virtual reality and augmented reality researchers. He conceived art of interactivity as opposed to the interactive art: there was more interest in interactivity design than in the art itself. [8] is one of his most representative works. The current paper will present an interactive installation where the body of visitors becomes the Interaction Device. Some examples are documented in literature, like [9] and [10]. Moreover, works on interactive floors with which the body is the only interaction device [11, 12] are documented too. The goal of the installation is to provide the user with the immersive experience of a soundwalk simulating an imaginary navigation inside the painting. The environment is formed by a rich sound background. Such outcome is realized by the painting sonification and the rendering of the users' footsteps over a bed of dry leaves, like the one represented in the painting of Figure 1a. Both the sonic background and the steps sound change according to the user's distance from the painting, which results in a sort of three-dimensional spatial development from the original two-dimensional artifact.

1.1 The Interactive Painting Sonification Project

This work proposal comes directly from the South Tyroler artist Hartwig Thaler [1] who has commissioned the Sound

[1] Further information about painter Hartwig Thaler can be found at his website https://www.hartwigthaler.de/

and Music Computing Group of the University of Padova [2] to produce a sonification of one of his works. The artist invited us to choose one painting among a series of 9, where various leaves textures are depicted. Although the textures would vary for density, leaf dimension and color, the paintings are united by the presence of a single subject (a carpet of dry leaves) occupying uniformly their entire surface. This suggests that also the sound rendering should be made regular and constant although with some degree of variability inside. Moreover, the paintings seem to be the result of the superimposition of different layers representing leaves of similar color, as confirmed by the creative process declared by the artist himself. This suggests the possibility of decomposing the image in different color matrices and to consider the painting as a sort of three-dimensional object of various superimposed layers. This idea envisions that various layers should not simply be seen as one above the other on the surface of the painting but that they can be disposed in the space in front of it. Thus, the painting is stretched from the wall into the space, allowing a user to walk through its various layers. The resulting interaction space is depicted in Figure 2 where the user walks on a wooden runway positioned in front of the painting. The runway is subdivided into three sections, each corresponding to the imaginary projection of the color matrices which can act as triggering points of some changes in the audio output. Thus, the installation does not only qualify as a mere image sonification project, but becomes a truly interactive environment based on the spatial projection of the painting. The paper is structured as follows: Section 2 focuses on the sonification of the painting; Section 3 describes the general architecture of the installation; Section 4 describes in details the localization system used in the installation; Section 5 presents the results of data analysis collected from the elements of previous section. Finally, section 6 summarize the results of the work and outline any possible further steps.

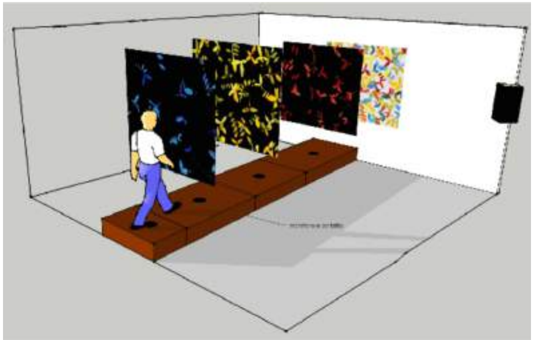


Figure 2: The painting sonification setup with the imaginary projection of three color matrices in the space in front of the painting.

2. SYSTEM ARCHITECTURE

The installation is characterized by two main elements: a wooden runway that leads the visitor toward the projection of the painting, and an audio reproduction system, fed by a sound background derived by the painting and users' step sonification. Fig.3 shows the schema of the installation, outlining the components of these two main elements. The soundscape of the installation changes according to the distance of the visitors from the image of the painting. In order to detect the position on the wooden runway, a system of eight piezoelectric sensors is used (Sensing). The signals acquired is then processed by means of an algorithm developed ad hoc based on the TDOA approach (Localization). Other localization approaches were considered. One of them exploit computer vision algorithm. This was not considered because of issue in case of unmanaged environment with multiple people interacting in the installation. Furthermore, it was necessary to detect either the user position or his steps to sonorize them. In addition, the installation has been developed with the collaboration of company specialized in diagnosis of wooden board and analysis of propagation time of waves. This led to the development of this specific system.

The information acquired are used to control the soundscape. It is produced by a bank of oscillator, fed by pink noise. Its outputs (Soundscape reproduction) are differently weighted according to an algorithm of image processing that extracts the needed features (Image feature extraction) from the three color layers of the painting (color matrix extraction). Furthermore, the runway is provided with a led "stream" that changes its color according to the color matrix of the painting. This provide a visual feedback so as to outline different portions of the runway each of which corresponds a different color layer. Further details on the elements illustrated in Fig.3 will be described in the following sections.

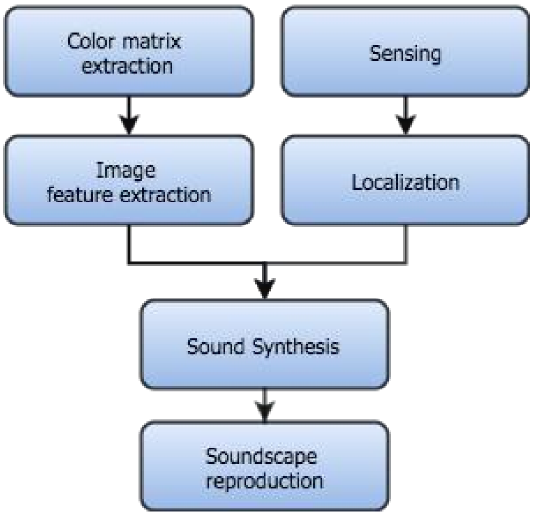


Figure 3: Schema of the system architecture: painting analysis, position detection and sonification

3. FROM THE PAINTING TO THE SOUND

The basic idea of this image sonification project is to consider the leaves as the elements of a mask, in which the colored spots represent open holes on a solid background.

[2] The website of the SMC group of the University of Padova is at `http://smc.dei.unipd.it/`

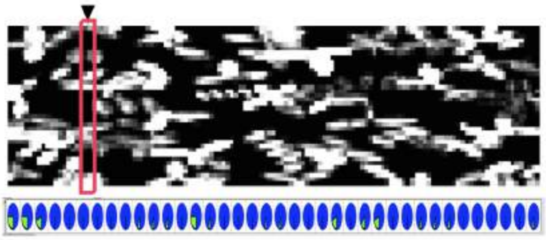


Figure 4: The 38x127 lightness matrix with the mixer controls of the separated filters output

		STEPS			
	SPECTRA	H	B	hB	lB
BACKGD.	H	▨			
	B		▨		
	hB			▨	
	lB				▨

Table 1: Combination possibilities of background and step sonification employing 4 different spectra: H (harmonic components), B (bell components), hB (highest range of bell components) lB (lowest range of bell components)

Transposing this interpretation in musical terms, the background comes to be a complex wide range spectrum of frequencies and the holes are the points where the single frequencies can be heard. In this perspective, the function of the image is similar to that of an orchestral score, where each line corresponds to a musical instrument or orchestral section. The black holes (signle note or group of notes) activate the section, while the white lines (rests) mute them. Hence, the spectrum plays the role of the whole orchestra and the image the role of the musical score. To realize this idea, we selected the painting with the lowest degree of leaves density among the 9 proposed (see Figure 1a). Following the idea of the various superimposed layers, we needed also to extract different colors matrices from the original painting. To do so we analyzed the image using the hue-saturation-value color model, where hue refers to the pure color, saturation to the quantity of white and value to the lightness [13]. Choosing an appropriate hue range, it is possible to obtain matrices that represent only the leaves of the selected color range. According to the suggestion of the artist, we extracted three matrices corresponding approximately to the primary colors (red, green and blue), shifting a little bit the green matrix towards the yellow which is the predominant color of the painting. The result of this process is depicted in Figure 1b, where a mask of yellow and green leaves extracted from the original image is shown.[3] To provide a rich background texture for the image sonification, we employed a bank of 38 band pass resonant filters[4] fed with pink noise and with separate controlled outputs. To superimpose the mask to the background texture, we scaled the yellow matrix in a 38 rows and 127 column matrix, where only the lightness values are reported (see Figure 4). This matrix can be considered as the real sonification score because it provides columns of lightness values which are scanned at a regular speed from the left to the right and vice versa. This timing mechanism allows the necessary link between the visual representation and the sequential nature of music. The time-controlled columns of lightness values are converted in numeric controls for a mixer matrix, thus making it possible to weight the filter's output according to the lightness of the every single pixel. This process results in different harmonies coming from the various weights and assigned to the spectral components. For instance, in the case represented in Figure 4, the marked column has higher values in the lower part of the painting and very small values in the higher part. This opens the lowest frequencies of the spectrum much more than the highest, allowing a major weight of the lowest components. The situations is very similar to a 38 voices chorale[5] where the dynamics of the various voices depend on the pixel lightness values.

3.1 The Installation Soundscape

The characteristic of the sonification engine described above is to be very close to the image features and to be easily adaptable to produce a painting-related soundscape. For this project we focused our attention on subtractive synthesis due to its flexibility and richness of expressive possibilities. The bank of filters can be fed with different input sounds and can be easily tuned according to various spectral models, which can be changed in relationship to the color of the matrix employed as musical score. Moreover, the resonant filters output allows a kind of smearing effect which smooths the harmonic transitions which characterize our image sonification approach, giving to the audio rendering a great variety in a seamless way. Another possibility offered by our sonification system is to implement a second bank of filters with the same characteristics of the first, being fed by a file reproducing the sound of a step on a carpet of leaves which can be triggered by impulses coming from a user's real step on the wooden runway. Thus, not only this would produce step sounds completely different from real ones, but these steps could be tuned according to a spectrum complementary to that used for the background. In the actual project we experimented four spectra: H (a harmonic spectrum of 38 frequencies starting from 100Hz), B (38 frequencies extracted from the full spectrum of a bell sound), hB (38 frequencies extracted from the highest band of the same bell spectrum) and lB (the same as above in the lowest band). Table 1 shows the full range of the possibilities in order to combine a background spectrum with its complementaries with a wide number

[3] Of course many other options are available to extract color matrices from an image, starting with the number of matrices to be extracted and with their color range. In our case the choice of extracting three matrices depends not only on the artist's suggestion but also on the actual available space for the interactive sonification on the sensor-equipped runaway (see Section **??**.

[4] The sonification is implemented in a Max/MSP patch employing the fffb object. See `https://docs.cycling74.com/max5/refpages/msp-ref/fffb~.html` for reference.

[5] The chorale is a vocal polyphonic composition, usually a religious hymn, typical of the sacred literature of the German Protestant Church under Martin Luther (1523). The chorale is characterized by a regular homo-rhythmic proceeding of the voices, exactly in the same way of the scanned columns of the image matrix.

of soundscape generation possibilities, which can be chosen according to the user's position on the sensor-equipped runway[6] .

4. THE SENSOR-EQUIPPED RUNWAY

The installation is composed of a wooden runway, with the dimension of 350x100x7cm, which guides the visitors towards the painting projection. The wooden board used for the runway has been individually chosen, in order to guarantee the most uniform possible speed of propagation of vibrations (see Tab.2). This feature is necessary in order to avoid difference in arrival time to the sensors, as far as possible. The measure of the speed of propagation in the boards was possible thanks to the tool Viscan [7]

# board	length (cm)	width (cm)	speed (m/s)
41	495	14,5	544,5
4	550	14,5	544,5
999	490	31	529,2
41	494	30	513,8
111	550	7,5	511,5
118	523	7,5	493,7

Table 2: Wooden boards description: the identifier number of the board, its length, its width and the speed of propagation of sound for each of them

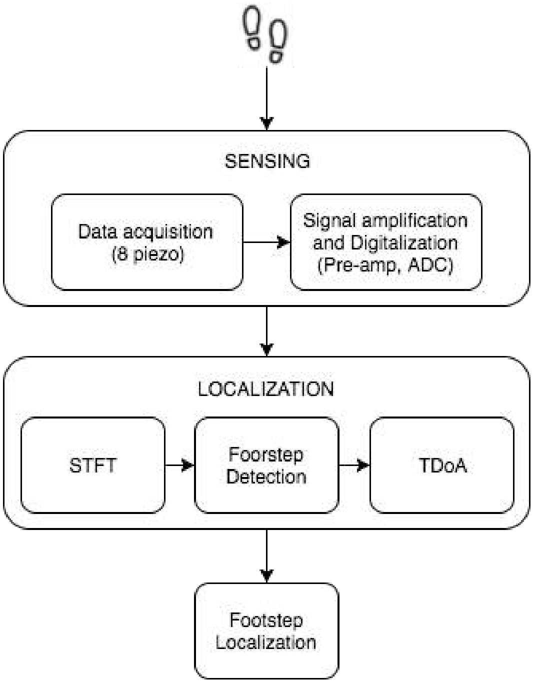

SENSING
Data acquisition (8 piezo)
Signal amplification and Digitalization (Pre-amp, ADC)
LOCALIZATION
STFT
Foorstep Detection
TDoA
Footstep Localization

Figure 5: Schema of the position detection system: Sensing is responsible of acquire the signals from the runway, Localization detect the steps and the position

The objective of sensing section is to acquire the footstep-induced structural vibration through sensors mounted on the runway. The sensing module consists of data acquisition, using eight piezoelectric sensors, signal amplification and digitization by means of a digital audio interface. The localization section is mainly divided in: STFT, footstep detection and TDOA estimation (see Fig. 5).

4.1 Footsteps characteristic

Human steps generates vibrations that propagate away from the source as seismic waves. For a single impact on an elastic half space, Miller and Purssey (1955) have shown that 70% of the energy of the impact is distributed in the Rayleigh wave. The remaining 30% of the energy is transmitted into the earth via body waves (transverse and longitudinal), while diminishing in amplitude as r^{-2} [14, 15]. The signal has also frequency dependent attenuation characteristics [16]. Furthermore, thanks to clinical gait analysis, like in [17–20], is possible to outline the GRF[8] of a footstep. Fig.6 describes the main behavior on amplitude/time graph, where the first peak is attributable to the heel strike and the second to the toe push-off as the body is propelled forward. In the light of these analysis, a “delay” time after the first peak, before next detection, was necessary in order to identify the position of the whole step at the heel position and to avoid continuous changes in short period of time. With this approach, the vibrations made by the step can be considered as produced by an impulsive hit.

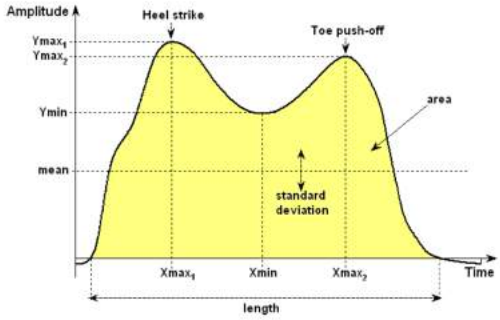


Figure 6: Representation of a usual Ground Reaction Force of footsteps on an Amplitude/Time graph

4.2 Sensing

Usual applications of floor vibrations detection use sophisticated sensors like the geophone sm-24. It can detect very small vibrations with high precision going to the detriment of very high price. In this application, due to the small size of the runway and the resonance of steps, a good SNR can be ensured with piezoelectric sensors of 2cm diameter. They are equally distributed on the runway in order to divide it in twelve areas, each of which is identified by the four nearest sensors that the localization system will use. As described in Section 3, the painting is divided in three color layers and each of which is assigned to the section of

[6] Some sound examples of these possibilities can be found at https://youtu.be/wzE17agkZ7I

[7] Description of this tool can be found at http://microtec.eu/en/catalogue/products/viscan/

[8] GRF (Ground Reaction Force) is, according to Newton's Law of Reaction, the force equal in magnitude but opposite in direction produced by the ground as reaction to the force the body exerts on the ground. The ground reaction force is used as propulsion to initiate and control the movement, and it is normally measured by force sensor plates.

the runway. This last is composed of four areas as outlined by the sensors described in Fig 7. Afterwards, in order to acquire the signal, the TASCAM US-2000 (8 Microphonic input) Digital converter was used. For this application, the audio interface works at 44.100kHz with a resolution of 24bit. The cable and connectors used to attach the sensors to the audio interface are respectively Klots and Neutrik so as to minimize possible filtering caused by unwanted parasitic capacitance.

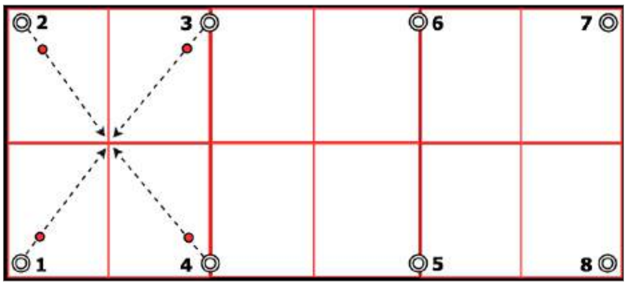


Figure 7: Disposition of the sensors and consequent division in twelve areas of the runway

4.3 Localization

Usually, the localization of sound events is based on the amplitude of time signals and the cross-correlation is used to find the lag between them [21, 22]. After several tests, this approach turned out to be unable to extract the necessary information because of the nature of the runway: wood is flexible and inhomogeneous, this lead to echoes and creaks that alter either the sound signal or speed. On the contrary, by means of STFT it is possible to clearly detect the onset of the footstep and so to highlight the TDOA. For the footstep detection, a moving average method is used: with two windows of size N and M, with $N >> M$ and N multiple of M, the average energy of the signal in the long period and in the short period are computed respectively. Every N samples the long period average is set, than every M samples a value of energy is computed and compared with the previous average energy (see Fig. 8). The algorithm was first simulated and evaluated using MATLAB with data sets of measured taken from the prototype runway equipped with four sensors. In formula:

$$\frac{1}{M}\sum_{m=0}^{M}|X(jN+kM+m)| > \alpha\frac{1}{N}\sum_{n=0}^{N}|X((j-1)N+n)|$$

with $X(f)$ is the Fourier transform of input signal $x(t)$, j number of the N-samples windows, k number of the M-sample windows and α a tuning factor. The tuning factor is necessary in order to make the system more robust against environmental noises. Additionally, in view of increasing the SNR of the signal, a cubic factor is applied to signal values.

Multilateration is a common approach for source localization that uses the time difference of arrival. Mathematically, the localization procedure can be described as

$$||x - p_2||_2 = v(t_2 - t_1) + ||x - p_1|| \qquad (1)$$

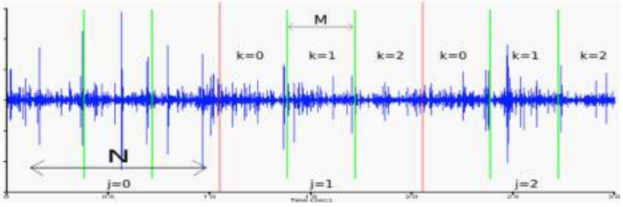


Figure 8: Graphical representation of the moving average algorithm implemented for the localization algorithm.

where p_1 is the location of the first sensor and p_2 is the location of the second sensor, x is the location of the excitation (i.e. footstep) and v is the propagation speed. This approach, view that non homogeneous material does not ensure constant speed of propagation of waves [23], can not be used. This is the reason why the runway is divided in twelve regions, so as that the maximum localization precision is proportional to the dimensions of each area. The information about TDOA of each sensor is finally processed, using a decision algorithm, to identify the region in which the event occurred. This algorithm is described in the next sections in the light of the results obtained from the evaluation test on the runway.

4.4 Real-time implementation

The localization system works in real-time thanks to the implementation of a software dedicated to the acquisition from Tascam buffer and the processing of the data received. The framework is implemented in C++ in order to maximize portability on different systems. A well-known and widely used library for managing Audio Interface buffers is used: Port-Audio. It provides a very simple API real-time streaming audio using a simple callback function or a blocking read/write interface. Finally, a visual interface using Qt simply allow to interact with the sensing and localization system. Qt is used for development multi-platform applications and graphical user interfaces (GUIs). This interface, during test phase, allows to visualize in real-time the position of the user on the runway and simply change algorithm parameters so as to test its correctness.

5. EVALUATION

At the beginning, in order to test the algorithm of step detection and to evaluate the results, a prototype of the runway was deemed necessary. A wooden pallet 120x80x15cm with a 2 cm thick wooden board whose base measures the same as height was used. The results obtained by the step detection algorithm on the prototype, shown in Fig.9, are: after 10 campaigns of data collection each of which composed of 16 footsteps, the accuracy measured is 91%. After this preliminary part, the next test was performed on the wooden runway. First of all, to analyze its response, the sensors have been positioned in the first four position of Fig.7 like for the test on the prototype. Afterwards, an experimental set-up was designed to make the analysis easily repeatable. In order to realize an homogeneous excitation from which extract the TDOA data, the experimental set-

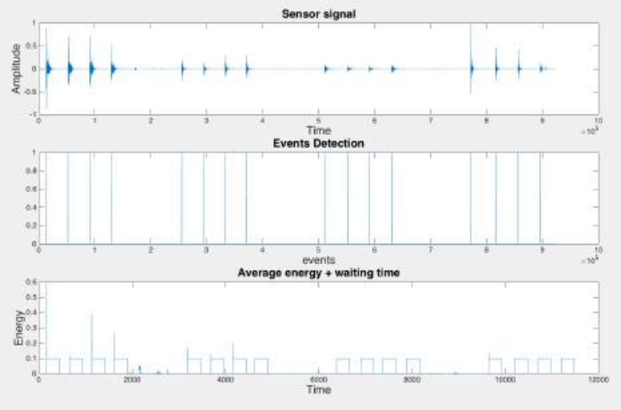


Figure 9: Event detection: First image shows the signal of sixteen steps, the second the detection of the steps and the third the average energy variation (straight line describes the waiting time after an event detection)

up was mainly composed of a 1 meter height reference and a mass of 50g. Along the diagonal, at 20 cm from each sensor toward the center of the relative area (red dots in Fig.7), 10 campaigns of series of 10 impulse excitations have been carried out using the mass dropped from the height reference. These two elements of the experimental set-up are respectively an adjustable stand for speakers, with a transverse metal bar, and a tennis ball. The choice of using a bouncing mass is dictate by the necessity to generate an impulsive excitation, therefore to avoid further bounces. This was possible taking the mass, while it was going, before the second bounce (see 10). The data collected from the sensors were then processed and the TDOA (measured in number of windows of size M) evaluated. Focusing on the impulses generated in the areas of sensor 1 and 2:

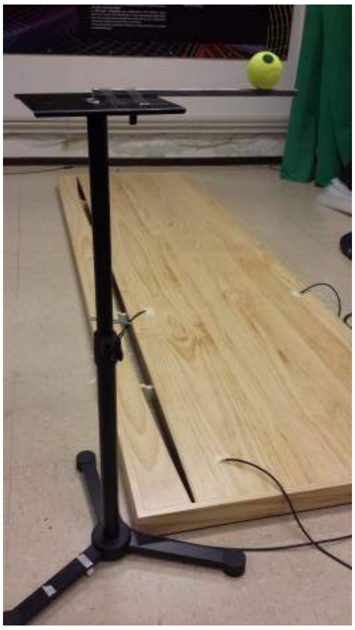


Figure 10: Experimental set-up: an adjustable stand for speakers, with a transverse metal bar, and a tennis ball

- The first sensor able to detect the hit is the nearest;
- The second sensor is the nearest to the first (i.e. sensor 2 if impulse is generated near sensor 1 and vice versa);
- The third and the fourth are respectively the one on the longitudinal direction and the one on the diagonal one.

An example of data is represented in Fig.11a where is shown a graph of the trend of TDOA between different hits in the same campaign. Therefore, these results are in contrast with those collected from sensor 3 and 4:

- The first sensor able to detect the hit is the nearest, as previous test;
- The second sensor is no more the nearest to the first but the one on the longitudinal direction.

These results are shown in Fig.11b and can be explained by the construction design of the runway: as explained in [23] the propagation of waves along the wood grain is faster than the transversal one. In the case of position 1, the two sensors are near the board of the runway, realized with a single peace of wood, which grain are horizontal with respect to long edge of the runway. Therefore, in this board the waves travel faster than the usual transversal propagation and reach the sensor in position 2 before the one in the position 3. On the contrary, sensor 3 and 4 are separated by wooden boards whose grain are transversal w.r.t. the abstracted joining line of the two sensor and this contrast the fast direct propagation of the waves.

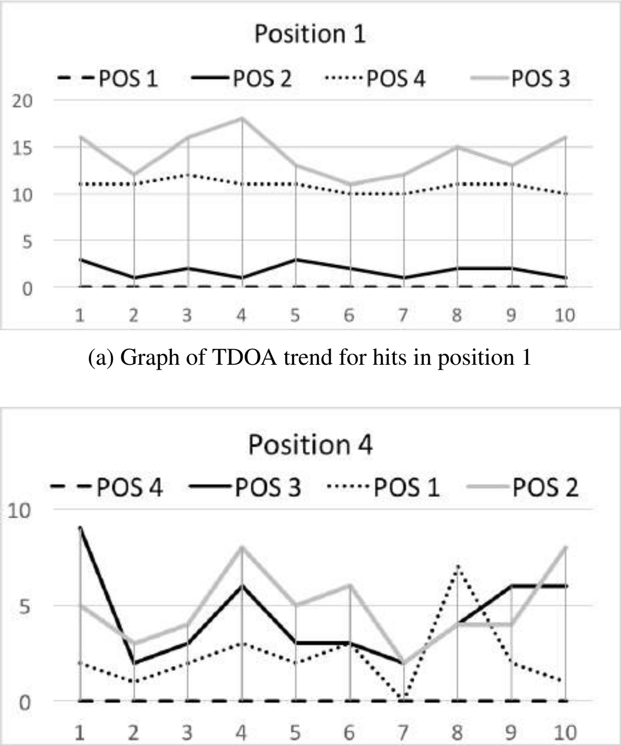
The mean and the standard error of the data collected for all the four sensors of the example campaign just described are summarized in Tab.3. They dictate the implementation of the decision algorithm that process the TDOA information according to which sensor is the first to detect an event:

- if the first is on the smaller side of the runway, the area where the event occurred is the same of the sensor position;
- if the first is on the longer side of the runway, the area is determined by the second sensor reached (i.e. if it is the number 6, than the area will be the one on the right of sensor 3 and vice versa with sensor 2).

Finally, in the light of these results, was possible to implement the decision algorithm and complete the Localization system for the wooden runway described in this article. Now, the installation can localize the position of the people on the wooden runway, change the parameters of the sonification algorithm and allow to discover a 2D painting in a new 3D approach.

6. CONCLUSIONS

The result of the project is the realization of an interactive installation which reproduce an auditory feedback according to the analysis either of the color features of the painting or the steps of the visitors on the wooden runway. The latter are detected thanks to a Localization system based on



(a) Graph of TDOA trend for hits in position 1

(b) Graph of TDOA trend for hits in position 4

Figure 11: Graphs of the TDOA trend (measured in windows of size N as in Fig.8) between signal sensors for excitations respectively in position 1 and 4: POS# describe the area of the sensor, the POS order in the legend is the expected order of arrival of waves at the sensors (geometric considerations); y axis describe the TDOA in number of windows and finally x axis describe the number of hit of a single campaign

the the TDOA measured by piezoelectric sensors, focusing on the first able to detects the steps.

One possible further step could be the use of machine learning techniques in order to map the position on the runway by means of a single sensor and studying the different features that can describe it. Some of these techniques have already been used for indoor localization and space mapping using microphones. With this application, this type of study could allow to overcome the non uniform speed of propagation of the wood. Furthermore this new approach could avoid interferences from outside the runway and from wood crackles and echoes, due to users' steps. A second likely further step could be the use of different sensors. One example could be microphones arrays, positioned just under the runway board. This system could recognize the direction of incoming sound waves and discriminate between the directions of interest and the one outside our range of interest. In the near future, in order to save the installation at the end of the exhibition, a preservation strategy has to be implemented, creating a digital copy and transposing the concepts of active preservation used for audio documents in the field of interactive installations [24–26].

Mean±S.E.	Sensor1	Sensor2	Sensor3	Sensor4
POS1	0	1.8±0.2	10.8±0.1	14.2±0.4
POS2	2.2±0.3	0	7.1±0.1	18.5±0.3
POS3	6.5±0.1	4.2±0.3	0	6.8±0.1
POS4	2.3±0.1	4.9±0.1	4.4±1.0	0

Table 3: Mean and standard error of the TDOA of a campaign of measure

Acknowledgments

This work has been partially supported by Microtec s.r.l. (Grant n. 98/2015).

7. REFERENCES

[1] L. Bullivant, *Responsive Environments: Architecture, Art and Design.* London: V & A Publications, 2006.

[2] M. Eisenberg, N. Elumeze, G. B. L. Buechley, S. Hendrix, and A. Eisenberg, "The homespun museum: computers, fabrication, and the design of personalized exhibits," in *In Proceedings of the 5th Conference on Creativity & Cognition*, ACM, New York, NY, 2005, pp. 13–21.

[3] E. Hornecker and M. Stifter, "Learning from interactive museum installations about interaction design for public settings," in *In Proc. of the 20th Conference of the Computer-Human interaction*, OZCHI '06, vol. 206. ACM, New York, NY, 2005, pp. 135–142.

[4] L. Terrenghi and A. Zimmermann, "Tailored audio augmented environments for museums," in *In Proceedings of the 9th international Conference on intelligent User interfaces*, IUI '04. ACM, New York, NY, 334-336., 2004, pp. 334–336.

[5] O. Bimber, F. Coriand, A. Kleppe, E. Bruns, S. Zollmann, and T. Langlotz, "Superimposing pictorial artwork with projected imagery," in *In ACM SIGGRAPH 2006 Courses*, SIGGRAPH '06. ACM, New York, NY, 2006.

[6] F. Avanzini, S. Canazza, G. D. Poli, C. Fantozzi, E. Micheloni, N. Pretto, A. Rodá, S. Gasparotto, and G. Salemmi, "Virtual reconstruction of an ancient greek pan flute," in *In Proc. Int. Conf. Sound and Music Computing (SMC 2016)*, 2016.

[7] F. Avanzini, S. Canazza, G. D. Poli, C. Fantozzi, N. Pretto, A. Rodá, I. Angelini, and C. Bettin, "Archaeology and virtual acoustics. a pan flute from ancient egypt," in *In Proc. Int. Conf. Sound and Music Computing (SMC 2015)*, 2015.

[8] M. W. Krueger, T. Gionfriddo, and K. Hinrichsen, "Videoplace — an artificial reality," in *in Proceeding of the SIGCHI Conference on Human Factors in Computing Systems*, 1985, pp. 35–40.

[9] F. Sparacino, “Scenographies of the past and museums of the future: from the wunderkammer to body-driven interactive narrative spaces,” in *In Proceedings of the 12th Annual ACM international Conference on Multimedia*, 2004, pp. 72–79.

[10] A. Webb, A. Kerne, E. Koh, P. Joshi, T. Park, and R. Graeber, “Choreographic buttons: promoting social interaction through human movement and clear affordances,” in *In Proceedings of the 14th Annual ACM international Conference on Multimedia*, 2006, pp. 451–460.

[11] K. Gronbaek, O. Iversen, K. Kortbek, K. Nielsen, and L. Aagaard, “igamefloor - a platform for co-located collaborative games,” in *In Proceedings of the International Conference on Advances in Computer Entertainment ACE*, Salzburg, Austria, 2007, pp. 64–71.

[12] K. Gronbaek, O. Iversen, K. Kortbek, K. Nielsen, and L. Aagard, “Interactive floor support for kinesthetic interaction in children learning environments,” in *In Proc. of INTERACT*, Rio de Janeiro, Brazil, 2007.

[13] M. D. Fairchild, *Color appearance models*. John Wiley & Sons, 2013.

[14] G.Succi, G. Prado, R. Gampert, T. Pedersen, and H. Dhaliwa, “Problems in seismic detection and tracking,” in *in Proc. of SPIE*, vol. 404, 2000, p. 165.

[15] G. Succi, D. Clapp, R. Gampert, and G. Prado, “Footstep detection and tracking,” in *in Proc. of SPIE*, vol. 4393, 2001, p. 22.

[16] H. Amick, “A frequency-dependent soil propagation model,” in *in Proc. of SPIE*, 1999.

[17] R. Vera-Rodriguez, N. Evans, R. Lewis, B.Fauve, and J. Mason, “An experimental study on the feasibility of footsteps as a biometric,” in *in Proceedings of 15th European Signal Processing Conference (EUSIPCO’07)*, 2007, p. 748–752.

[18] S. Winiarski and A. Rutkowska-Kucharska, “Estimated ground reaction force in normal and pathological gait,” in *Acta of Bioengineering and Biomechanics*, vol. 11, no. 1, 2009.

[19] M. Syczewska and T. Oberg, “Mechanical energy levels in respect to the center of mass of trunk segments during walking in healthy and stroke subjects,” in *Gait & Posture*, 2001, p. 131.

[20] V. Inman, H. Ralston, and F. Todd, in *Human walking*. Baltimore, London: Williams & Wilkins, 1981.

[21] D. Salvati and S. Canazza, “Incident signal power comparison for localization of concurrent multiple acoustic sources,” *The Scientific World Journal*, vol. 2014, p. 13 pages, 2014.

[22] D.Salvati and S. Canazza, “Adaptive time delay estimation using filter length constraints for source localization in reverberant acoustic environments,” *IEEE Signal Processing Letters*, vol. 20, no. 5, pp. 507–510, 2013.

[23] I. A. Veres and M. B. Sayir, “Wave propagation in a wooden bar,” *Ultrasonics*, 2004.

[24] S. Canazza, C. Fantozzi, and N. Pretto, “Accessing tape music documents on mobile devices,” *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 12, no. 1s, pp. 20:1–20:20, Oct. 2015. [Online]. Available: http://doi.acm.org/10.1145/2808200

[25] F. Bressan and S. Canazza, “The challenge of preserving interactive sound art: a multilevel approach,” *Int. J. of Arts and Technology*, vol. 7, no. 4, pp. 294 – 315, 2014.

[26] C. Fantozzi, F. Bressan, N. Pretto, and S. Canazza, “Tape music archives: from preservation to access,” *International Journal on Digital Libraries*, pp. 1–17, 2017. [Online]. Available: http://dx.doi.org/10.1007/s00799-017-0208-8

Interactive Soundscapes: Developing a Physical Space Augmented through Dynamic Sound Rendering and Granular Synthesis

Matteo Lionello
Sound and Music Computing
Aalborg University Copenhagen (Denmark)
mlionel16@student.aau.dk

Marcella Mandanici, Sergio Canazza, Edoardo Micheloni
Dept. of Information Engineering
University of Padova (Italy)
surname@dei.unipd.it

ABSTRACT

This project presents the development of an interactive environment inspired by soundscape composition, in which a user can explore a sound augmented reality related to a real soundscape. The user walks on the application's responsive floor controlling through her/his position the soundscape rendering. In the same time s/he can explore the sound structure entering central zone of the responsive floor, where some granular synthesis processes are activated. One main goal of the project is to sensitize the participants regarding the surrounding sonic environment looking also for pedagogical studies. A first evaluation of an audio excerpt produced by a user's soundwalk shows that the application can produce good quality and interesting soundscapes, which are fully consistent with the real environments from which they were inspired.

1. INTRODUCTION

Every day, in any place, we are all surrounded by an eternal and universal symphony. It is eternal because, in John Cage terms, as long as there is a person able to hear, there always will be a sound to be heard. The music is already around us, and we are immersed in it. What is the role of the artist in front of this already done music? Are we only passive listeners with no control over it or are we "*responsible for giving it form and beauty?*" [1]. This question has been written at the beginning of "*Tuning of the world*" by Murray Schafer, where he gave a definition of the term "*Soundscape*" indicating the set of sounds persisting into a certain context. This established the basis of the Soundscape Composition which will be afterwards carried on by Barry Truax, colleague and successor to Murray Schafer. Barry Truax deeply studied the granular synthesis and composed musical pieces implementing it in real time through electronic oscillators, sample sounds of small size, and finally using soundscape recordings. Soundscape composition is a form of electroacoustic music, characterized by recognizable contexts and environmental sounds. The original environmental context is preserved: all the memories, past experiences and associations are called to the listener's mind through artist's compositional strategies [2]. The artist is the medium between the soundscape and the listener. The aim of the artist is to invoke in the listener associations, memories and fancies referred to a particular soundscape.

1.1 Active and Deep Listening

Two different concepts have been strongly remarked by Francois Pachet and Pauline Oliveros. The first one from Pachet is held in the term "*Active Listening*" [3], which refers to the capability of a user to have an active and concrete control on what s/he listens to. The second concept from Oliveros is held in the expression "*Deep Listening*" [4], which outlines the concentration of the audience and the musicians themselves on the meaning beyond the sounds that are perceived and played. Both the concepts are basic and strongly present in the installation that we introduce here. The work presented in this paper is inspired by Barry Truax's "*Pacific Fanfare, Entrance to the Harbor*" [5] where the listener is accompanied by Truax in a sound-route in the harbor of Vancouver. In the 1994 revision, Truax implemented the recorded soundscape with some compositional tools bringing the listener between two parallel worlds which follow each other during the track, one related to the real Vancouver harbor soundscape, and the other one from the musician's interpretation of it. In this kind of soundscape composition the listener is led by the composer on a fixed flow of acoustic impressions captured by the artist and both of them are involved in a deep listening of the soundscape in Oliveros' terms. Music interactive systems allow to extend the user's freedom exploring the sound augmented environment with no external guides but her/his own auditory feedback. In this way the listener is no longer considered as a passive audience but s/he can use some degree of control on the music, proposing different perceptions of what is reproduced using memory associative routes. Moreover, her/his interaction depends on the sound context the system reproduces, promoting a deep listening of the surrounding sonic events.

1.2 Interactive Soundscapes

Interactive Soundscape is a 3x4 mt. floor under the range of a camera. When the user enters the installation s/he is projected in a sonic environment composed by sounds related to a real environment[1]. The aims of this project are both the spatial exploration of the reconstructed soundscape and the structural exploration of the sound elements peculiar to that soundscape. Using a granular synthesis process, brief time-windows are extracted from the sounds composing the soundscape. The user changing her/his position inside the installation modifies the parameters applied to the processing of the sound sources. Thus, moving around it is possible to explore the internal structure of the sounds projecting the user into a world as much real as the original one, emphasizing the acoustic properties of each sound. The role of the artist takes a step back leaving the tools for the musical composition to an audience not necessarily musically educated and not even aware of them. Besides the artistic purposes, the installation can be used in pedagogical field looking how children manage the spatial information to study cognitive processes [6] [7]. When the user gets the feedback from the different sounds distributed over the augmented space, cognitive maps are built by abstraction processes according to them. Thus, when a user enters the tracked area s/he can use what has been previously learned, enriching her/his cognitive map of the surrounding space.

2. RELATED WORK

Spatial cognition is widely used in interactive installations such as soundscape rendering and audio augmented environments. The design of composed soundscape for virtual environments requires a formal description and organization of the sound events which are used in it. The distinction between background audio and sonic events has been well defined by the Music Technology Group at Pompeu Fabra Universidad in Barcelona. They developed an online platform "*Soundscape Modeling*" that widely describes the state of the art about the possible applications of soundscape rendering and design, providing a powerful tool for authoring process. An online platform facilitates the process and generation of realistic soundscapes, focusing especially on computer games and mobile and web applications. Sample research from `Freesound.org`[2] is simplified by an automatic audio classification of sound features to compose graph model related to particular taxonomies. The soundscape is composed according to a two layers based architecture. On the first layer all the background sounds are propagated, whereas on a upper layer there are dynamic sounds events with which the users are able to interact. In a server/client architecture, the server gets real-time information from the clients coordinates related to a space providing a web stream of the soundscape. Combining SuperCollider[3], Freesound and GoogleMaps it is possible to provide soundscapes formed by collage of sounds according to specific position in a virtual city-visit in GoogleMap[4]. Mobile equipped with GPS and Internet can be used to enrich the experience of walking through a city listening to a particular soundscape that is streamed by the application and guided by the client's position. Compared to "*Soundscape Modeling*" where the interaction is interfaced through GPS or relative position in a virtual world, detecting a user's interaction in a augmented reality environment can be a challenging topic. Responsive floor interactive systems are one of the main system used to detect the user's interaction. They are mostly implemented through computer vision, audio capture and tracking techniques. For the Swiss Expo in 2002, in Neuchatel has been developed a complex artistic installation called Ada [8], which consisted of an artificial organism capable to interact actively with its visitor capturing their interaction and replying using visual and sound activities. The aim of the project was to provide Ada of the ability of expressing its internal status, which varied according to the input from people's interaction. "Her" core was composed by a tracking system recording weight, direction, speed and location of the visitors walking on its responsive floor; a vision system which used cameras controlled by the localization of the users for visual information; and an audio system to "*help identify salient individuals*". Beside computer vision and microphones, another technique used for tracking the user inside an augmented environment is triangularization through wireless devices. The LISTEN [9] project supplies to museum and art exhibition's visitors an augmented and personalized visit. Through motion-tracked wireless headphones, users exploring the physical space get immersed in a virtual environment which augments the space itself. A soundscape is composed dynamically differently for each visitor according to how s/he moves inside the space, to previous collected information of his path and his preferences and interests gathered during her/his walk. The experience provided by the installation varies from person to person, according to their behavior in the environment, allowing thus a higher level of visitors engagement.

3. SYSTEM DESCRIPTION

Interactive Soundscape uses a set of four loudspeakers, one camera and a computer running the two parts of the software and connected to both the audio system and the camera. It is presented as an indoor space under the range of a camera placed at the top of the room with its optical axis perpendicular to the floor. The installation is an interactive system composed by a tracking system capable of producing a couple of Cartesian coordinates of a user. Her/his relative position is transferred via OSC[5] [10] to a module written in PureData[6] [11] to process the audio data. The user entering the tracked zone is virtually transposed into a certain sonic environment built around her/him. Ac-

[1] A video of the installation is available at `https://vimeo.com/146137215`.

[2] Freesound.org is a collaborative on-line sound database, providing meta-data and geo-localization data for the uploaded sounds. `http://freesound.org`

[3] Supercollider is a developing environment and programming language used for real time audio synthesis. `http://supercollider.github.io/`

[4] Google Maps is a web mapping service developed by Google, including satellite images, topographic city and rural maps, and 360 degree street view perspective `https://maps.google.com/`

[5] Open Sound Control is a networking protocol developed at CNMAT by Adrian Freed and Matt Wright for musical instrument controls to comunicate in a network

[6] PureData is a visual programming language Open Source developed by Miller Puckette, more information can be found at `https://puredata.info/`

cording to the user's real location her/his virtual position is modified as well and, consequently, the processing of the audio data is differently shaped. The result is a soundwalk obtained by an acoustically augmented reality.

3.1 The Soundscape's Design

While the granularization technique is traditionally used on a physical fixed support, this installation projects is a real-time result of the interaction of a user with a physical environment. The human-machine interaction takes place through the spatial cognition of the user who decides to move according to her/his spatial perception combined with her/his auditory memory of the areas which s/he has already explored. As a tribute to the work "*Pacific Fanfare*" by Barry Truax, for testing the installation it has been decided to propose a sound environment similar to the one played by the artist: a harbor soundscape.

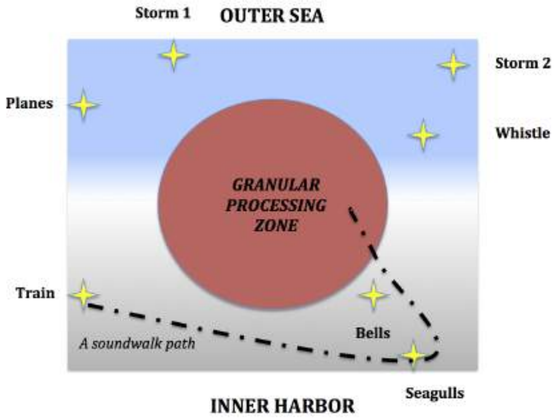


Figure 1. Graphic representation of the sound events positioning on the Interactive Soundscapes active area, with a user's soundwalk path.

In order to reproduce this soundscape, train and plane sounds were used to simulate the harbor industrial area, seagulls and bells sounds for the docks, weaves, boats whistles and far storms to simulate the farthest zone of the docks to the sea, and human activity sounds to simulate the inner land of the harbor.[7] The waves and the human activity sounds are used as the background of the soundscape and are processed with the granular synthesis in the central circular area shown in Figure 1, while the other files are used as events (some of them processed with reverberation and other with granular synthesis). Other configurations for different soundscapes are possible. Starting from a set of audio files for a particular context, a supervisor can load them in the PureData patch where the virtual location of the audio sources is decided setting a pair of coordinates. Those coordinates refer to the location where the user will hear the maximal amplification from that audio. Likewise it can be set the type of decreasing amplitude as the user goes further from an audio source. The granular zone is set choosing the circle radium and center over which the granular synthesis takes place. It's necessary to select the audio files to use for the granular synthesis, whereas files not selected files will be processed with a deep reverb. By walking inside and outside boundary distinguishing the two different areas, a 1400 ms crossfade between the two flows - one made from granular synthesis and the other one from sum of the amplitude of the audio sources - is applied.

3.2 The Design of the Parameters

The source location for the audio files can be chosen to be inside as well as outside the augmented space. In the case the coordinates are set outside, the user will never be able to reach that source and the sound will never gain the maximal intensity. The synthesis parameters depend on the distance between the user and the center of the granular zone. Along the edge the grains are 100 ms long, the density is maximized (100% of probability for two grains to succeed), the parameters for the band pass filter applied to the grains uses low cut off frequencies between 150 and 300 Hz, whereas the quality factor's values are ranging between 5 and 20, and the variance parameter describing the heterogeneity in the granular streams modeling is set to 0.1 in order to model all the streams around a narrow neighborhood of the values set by the user's position. As the user goes closer to the center of the granular zone, the soundscape results more and more chaotic: in the center the duration of the grains is set equal to 10 ms, the density is 25% and the cut off frequencies and the quality factors reach high values. The granularization happens in real time during the playback of the audio extracting every grain in each stream one after one from the original audio file. In Figure 1 it is possible to see a representation of the composed soundscape and the spatial organization of sounds inside the augmented space. The squared area represents the area covered by the camera, the red area is where the granular synthesis occurs, the blue and gray shaded backgrounds starting from the top and the bottom of the figure represent the amplitudes of the two different background sounds which are mixed together. The diamonds refer to the events composing the soundscape on the background layer.

4. SYSTEM ARCHITECTURE

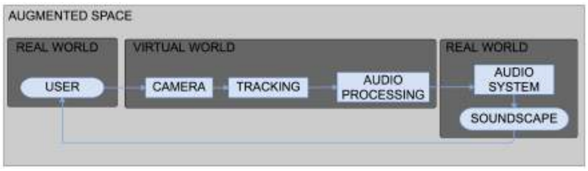


Figure 2. Flow chart of the system architecture of Interactive Soundscapes

The system architecture is made of two different software modules (Fig. 2). The first module is designed to elaborate the images captured by a camera, oriented perpendicularly respect to the floor. The camera covers a surface of variable dimension depending on the height at which it is

[7] All the audio files have been selected from `http://freesound.org`

positioned. A software named *Zone Tracker* [12] elaborates the visual information, subtracting the instantaneous images from a dry image of the ground. From the resulting image, calculated by averages of morphological transformation, a "*blob*" is obtained and its barycenter represents the user position in the area covered by the camera. The coordinates of the user's barycenter are sent, through OSC, to a second software module apt to the interaction sound processing. The second module is written in PureData, which processes the user's position with the loaded audio files, creating for each file an audio stream scaled in amplitude in function of the user position. A second stream is as well created for each file, performing a granular synthesis of the first one.

4.1 Granular synthesis

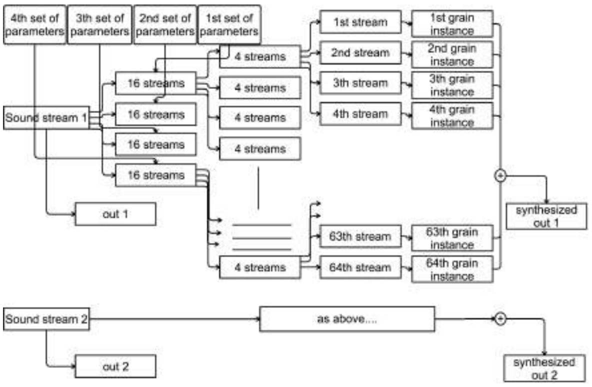


Figure 3. Granular streams architecture for two different audio sources. Each original audio input is split into 4 different classes of streams characterized by different granular parameters. The resulting asynchronous granular streams are summed together into a synthesized output stream.

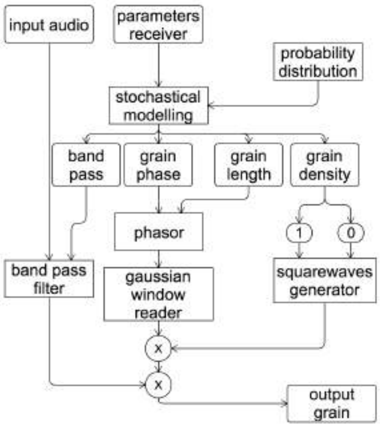


Figure 4. Grain structure. Each of the 64 audio streams for each audio source are filtered by a band pass filter depending on the user's position and enveloped according to the input parameters.

The streams of grains is obtained imposing to one audio a train of windows function of short length. Granular synthesis has been designed to operate on individual audio stream from each source. In order to obtain 64 granular sub-streams for each source, the main stream has been recursively divided in 4 sub-stream for 3 times as shown in Figure 3. To make the final stream richest and as various as possible, each of the first 4 sub-streams (see Figure 4) are defined with a set of parameters controlling duration, density, mid-frequency and Q factor of the granular stream. Each of these parameters are then stochastically modeled depending to a dynamic variance parameter. All the 64 sub-streams are then summed in one output resulting stream. Even more the resulting streams relating to each source are directed in one last output audio flow. To develop in a various and plentiful way each of the 4 subflows sharing the same set of control parameters, they have been modeled through a probability distribution such that there could be no two sub-streams identical in any aspect.

5. EVALUATION

To provide a first evaluation of the soundscape generated in the Interactive Soundscape environment we prepared an online listening test with the following aims:

1. to assess if the parameters of the audio engine which produce the computer generated soundscape respond to the perceptual expectations of the listeners
2. to evaluate how listeners perceive the computer generated soundscape
3. to see how listeners link the computer generated soundscape to a real environment.

Thirty-one invited listeners took part in the test, 25 males and 6 females. Subjects were aged between 20 and 57 years (mean = 31.5 years, *SD* = 8.8). Seventeen subjects have studied a musical instrument for more than 5 years (54.8%), nine have studied a musical instrument for less then 5 years (29%), and five have never studied a musical instrument (16.1%). For the test we selected an audio excerpt recorded during an exploratory session performed by a single user, whose soundwalk path is depicted in Figure 1. The excerpt lasts approximately 1 min and 30 s and comprehends both soundscape and granular synthesis produced sounds, with granular processing beginning at s16. Figure 5 reports a rough reconstruction of the event appearance and timing in the first 21s of the audio excerpt chosen for user's evaluation. In this representation at least five events are detected before the user enters the granular processing zone (waves, train, birds, bell and seagulls) [8]. The audio excerpt is not a simple soundscape recording but is a computer generated soundscape which refers to a real soundscape but which is not real. Beyond reverberation effects, it contains sounds obtained through a granular synthesis process which can significantly affect the perception of the user. Thus, to assess if the parameters of the audio

[8] This interpretation is not unique. Due to the complexity of the soundscape recordings employed and to the differences in the listening conditions (which in an online test cannot be easily controlled), different listeners may report different events or a different event order.

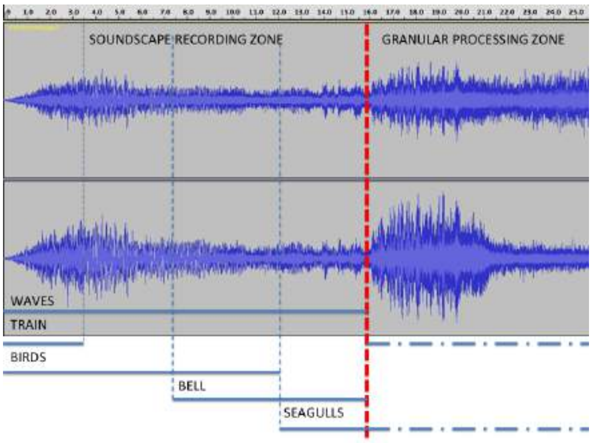


Figure 5. Graphic representation and event timing in the first 21s of the audio excerpt chosen for user's evaluation.

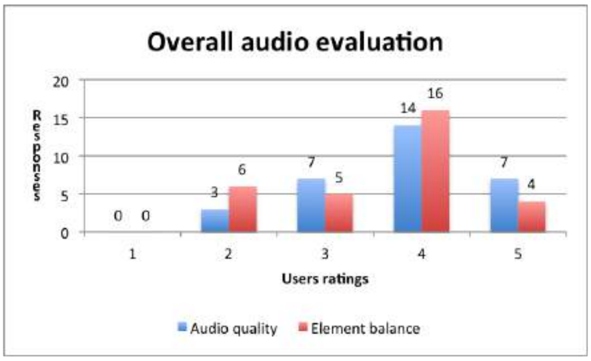


Figure 6. Histogram of users ratings for audio quality (in blue) and soundscape elements balance (in red), on a range from 1 (very bad) to 5 (very good).

engine respond to the perceptual expectations of the listeners, we asked our subjects to assign a rating ranging from 1 (very bad) to 5 (very good) to the audio quality and to the soundscape elements balance of the audio excerpt [9]. Results reported in Figure 6 show that the good rating (4) is the group which collected the majority of listeners responses (45.2% for audio quality and 51.6% for element's balance).

For the second part of our test we relied on the method described in [13], which identify *pleasantness* and *eventfulness* as the two main components for soundscape evaluation. Adapting the circumflex model of affect described by [14] to the soundscapes qualities, the authors define a circular two dimensional space which includes eight soundscape descriptors: eventful, exciting, pleasant, calm, uneventful, monotonous, unpleasant and chaotic. Thus, we asked our listeners to evaluate the *pleasantness* and *eventfulness* of the computer generated soundscape on a scale ranging from 1 to 5 in order to build a perceptual map of how listeners perceive the computer generated environment. The results are reported in Figure 7. The major number of positions are concentrated in the *eventful–exciting–pleasant* quarter of the perceptual map area and collect an

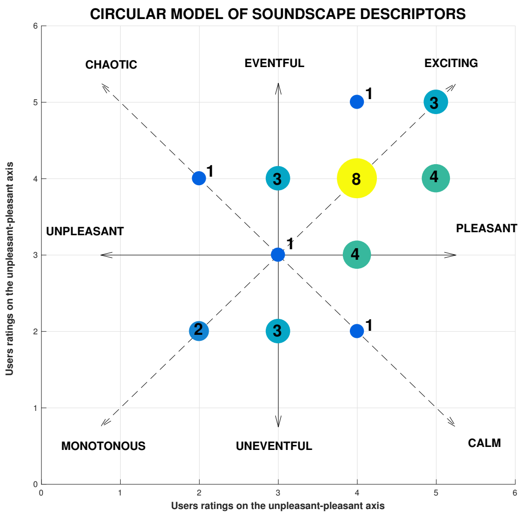


Figure 7. The perceptual map resulting from the ratings of the participants to the computer generated soundscape test. The numbers near the circles show the collected responses for each position.

overall of 24 responses. The greatest group (in yellow, with 8 responses) stands at point 4 on the *monotonous–exciting* diagonal axis.

In the version of Interactive Soundscapes employed for the test we reproduced a harbor soundscape through the use of many sound recordings. The result is a complex and multifaceted soundscape, which offers to listeners various aspects of the environment and which could also lead to different interpretations. To see how listeners linked the audio excerpt to a real environment, we asked our subjects to describe at least one physical place that could have created such a soundscape. A textual analysis of the answers [10] reports that the top phrase containing 2 words is "train station" with 6 occurrences. The single word with the major number of occurrences is "harbor" (11) followed by "station" (7), "train" (6), "port" (4) and dock (4). Words referred to natural places or sounds such as "beach", "rain", "foggy" and "sea" report only 1 occurrence. No occurrence is reported for seagull or bird sounds.

6. CONCLUSION

More than a half of the test subjects assigned a good or very good rating to the audio quality and element balance of the computer generated soundscape. This finding, together with the fact that more than 77% of the test subjects assigned an evaluation between 4 and 5 in the eventful-exciting-pleasant zone of the circular map of soundscape descriptors, seem to indicate that this implementation of the Interactive Soundscape application can generate sonic experiences of good quality. Moreover, the soundscape is more exciting than monotonous and more pleasant than unpleasant, and this is a result which could not be given for

[9] The complete Likert scale is 1(very bad), 2 (bad), 3 (medium), 4 (good) and 5 (very good).

[10] The online tool used can be found at `https://www.online-utility.org/text/analyzer.jsp`

granted in advance considering that the audio excerpt is the product of a soundwalk in a virtual environment and not the reproduction of a real soundscape. Due to methodological and technical difficulties it was not possible to assess if subjects could perceive the sound changes obtained through the granular synthesis and how they evaluated them. This is an important task which requires a strict control on the listening conditions and a major control on the audio file's content. As a matter of facts, the presence of a great number of different sounds, both in the foreground and in the background, can confuse the listeners and does not guarantee a sufficient sound discrimination. No textual description of the physical place which could have generated the soundscape was delivered to the test subjects. They spontaneously identified it mainly as "train station" and "harbor", showing thus that the sound rendering is recognizable as a real soundscape and that it is consistent with what we wanted to reproduce.

6.1 Further work

The next steps for this work will be to collocate it in a 3D audio context using a wave-field synthesizer and providing the system to detect interactions from more people at the same time. The sound information will be local instead of being the same in the whole area, and the parameters collected from the participants will be used only for the manipulation of the dynamical events over the soundscape.

Acknowledgments

We want to thank Leonardo Amico for his support in programming the *Zone Tracker* application.

7. REFERENCES

[1] R. M. Schafer, *The soundscape: Our sonic environment and the tuning of the world.* Inner Traditions/Bear & Co, 1993.

[2] B. Truax, "Soundscape composition as global music: Electroacoustic music as soundscape," *Organised Sound*, vol. 13, no. 2, pp. 103–109, August 2008.

[3] F. Pachet, "Active listening: What is in the air?" Sony CSL, Tech. Rep., 1999.

[4] P. Oliveros, *Deep listening: a composer's sound practice.* IUniverse, 2005.

[5] (1999). [Online]. Available: http://www.sfu.ca/~truax/vanscape.html

[6] A. Camurri, S. Canazza, C. Canepa, G. L. Foresti, A. Rodà, G. Volpe, and S. Zanolla, "The stanza logomotoria: an interactive environment for learning and communication," in *Proceedings of SMC Conference 2010, Barcelona*, 2010.

[7] M. Mandanici, A. Rodà, and S. Canazza, "The harmonic walk: an interactive physical environment to learn tonal melody accompaniment," *Advances in Multimedia*, vol. 2016, p. 2, 2016.

[8] K. Eng, A. Babler, U. Bernardet, M. Blanchard, M. Costa, T. Delbruck, R. J. Douglas, K. Hepp, D. Klein, J. Manzolli *et al.*, "Ada-intelligent space: an artificial creature for the swissexpo. 02," in *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, vol. 3. IEEE, 2003, pp. 4154–4159.

[9] G. Eckel, "Immersive audio-augmented environments," in *Proceedings Fifth International Conference on Information Visualisation*, 2001.

[10] M. Wright, "Open sound control-a new protocol for communicationg with sound synthesizers," in *Proceedings of the 1997 International Computer Music Conference*, 1997, pp. 101–104.

[11] M. Puckette, "Pure data: another integrated computer music environment," in *Proceedings of the second intercollege computer music concerts*, 1996, pp. 37–41. [Online]. Available: https://puredata.info/

[12] S. Zanolla, S. Canazza, A. Rodà, A. Camurri, and G. Volpe, "Entertaining listening by means of the Stanza Logo-Motoria: an Interactive Multimodal Environment," *Entertainment Computing*, vol. 2013, no. 4, pp. 213–220, 2013.

[13] Ö. Axelsson, M. E. Nilsson, and B. Berglund, "A principal components model of soundscape perception," *The Journal of the Acoustical Society of America*, vol. 128, no. 5, pp. 2836–2846, 2010.

[14] J. Posner, J. A. Russell, and B. S. Peterson, "The circumplex model of affect: An integrative approach to affective neuroscience, cognitive development, and psychopathology," *Development and psychopathology*, vol. 17, no. 03, pp. 715–734, 2005.

Learning to play the guitar at the age of interactive and collaborative Web technologies

Anne-Marie Burns, Sébastien Bel, Caroline Traube
Faculty of Music, University of Montreal
anne-marie.burns/sebastien.bel/caroline.traube@umontreal.ca

ABSTRACT

Recent advances in online technologies are changing the way we approach instrumental music education. The diversity of online music resources has increased through the availability of digital scores, video tutorials and music applications, creating the need for a cohesive, integrated learning experience. This article will first explore how different technological innovations have shaped music education and how their actual transposition in the digital world is creating new learning paradigms. As a practical illustration of these paradigms, we will present the development of a new online learning environment that allows master-apprentice or self-taught learning, using interactive and collaborative tools.

1. INTRODUCTION

In order to understand how technology can help a self-taught learner, or a learner in between lessons with a master, we must present the specificities of private and group lessons. In this introductory section, we will identify the learning processes in place and determine which are missing when the learner is left alone. We will then review how technology has been used to mitigate these in the past and how new Web technologies can be used today.

1.1 Group Music Classes vs Private Music Lessons

Table 1 describes the master and the learner activities. It identifies the modalities in place and the type of feedback that is possible between them. In the typical context of a private lesson, prior to any lesson, both protagonists will discuss the learner's intrinsic motivations. The master will adapt his teaching according to the learner objectives and prior knowledge. This differentiated teaching will continuously occur throughout the lesson series since the master will adapt his teaching to the learners progress. During a lesson, the learner observes and receives direct instructions from the master, but he also plays with his master. That allows him to regulate his action according to what he observes, ears and feels. The learner also has the ability to discuss his sensations with his master and to receive feedback on his action. All these interactions are believed to contribute to learning and to help to maintain motivation.

Similar activities can be observed in group lessons. In that context, there might be less emphasis on individual objectives and progression but a gain benefit on interaction with peers. In a group context, the attention of the master is shared between the learners. Learners are able to regulate their action with respect to the master, to the group and among themselves. Learners can discuss their sensations and request feedback from peers, and consequently form a community of practice.

1.2 Informal learning in-between lessons or in the case of self-taught learners

Table 2 presents the tasks of the learner in-between lessons (or in the case of a self-taught learner) and relates these to the learning and teaching activities presented in Table 1. It shows which technologies can be used to mitigate the absence of a master and of peers. For each activity, the table presents different types of technologies and gives a few examples of existing commercial and research solutions.

1.2.1 Search for Material

One of the tasks the learner might want to perform is to search for new material. A typical seft-taught learner will want to search for his favourite artists' songs. Websites to share tablatures exist on the Web for a long time. At first, this information was stored in simple text files using ASCII characters to represent strings and fret positions, techniques and chords information. The Ultimate-Guitar Website [16] (see Table 2 for all technologies' references) is one example of this type of sharing platforms. Nowadays, these sharing platforms have evolved and can display the score in standard western notation and advanced tablature notation. They also offer playback functionalities and permit, for example, to modify the playback tempo or to loop sections of the score (Songsterr [15] is one of these). The most elaborate interfaces will even link and play synchronously audio and video recordings of a performance (Soundslice [14], JellyNote [13]).

Music information retrieval algorithms offer new possibilities to extract playing instructions directly from a recording. There already exist some commercial software applications where one can input audio recording and

No.	Teaching activities	Learning activities	Modalities	Feedback
1	Ask the learner(s) about his / their goals and motivation. Perform a formative evaluation to determine group or individual level.	Tell the master about his / their goals and motivation. Perform the request evaluation.	Aural, visual, tactile	Discuss the feasibility of the goals and objectives. Help to establish attainable goals. Determine the necessary learning path and steps for the group or individual.
2	Prepare adapted material.	Receive the master lessons plan.	Aural, visual	Discuss the learning plan. Continuous adaptation to the group and to the individual's progression.
3	Explain a concept, an exercise or a musical excerpt.	Listen to explanation of a concept, an exercise or a musical excerpt.	Aural, visual	The master can give oral or written instructions (annotation on a score or in a lesson book) to the learners. Learner(s) can also take personal notes.
4	Perform a demonstration on the instrument.	Observe the master demonstration.	Aural, visual	The learner(s) can ask the master to repeat all or part of the demonstration. The learner(s) can ask for more information.
5	Ask the learner(s) to perform a musical excerpt or an exercise individually or in groups.	Play a musical excerpt or an exercise individually or in groups.	Aural, visual, tactile	The master can give oral or written instructions (annotation on a score or in a lesson book) to the learner(s). The master can physically correct the position and gesture of the learner(s). Learner(s) can listen and observe to other learners performance. They can regulate according to the other's performance. They can also discuss and comment peers performances.
6	Perform a musical excerpt or an exercise (including improvisation) along with the learner(s) individually or in groups.	Perform a musical excerpt or an exercise along with the master individually or in groups.	Aural, visual, tactile	Realtime regulation: the learner can adjust his performance to what he hears and observes from the master and from the other learners.
7	Ask learners to practice musical excerpts or exercises (including improvisation) in small groups.	Practice musical excerpts or exercises with peers.	Aural, visual, tactile	Realtime regulation: the learner can adjust his performance to what he hears and observes from the other learners. Learners can also discuss, comment and manipulate each other.

Table 1: Teaching and learning activities in private and group lessons.

retrieve an analysis of the chords and notes played (Chordify [17], Capo [18], GuitarMaster [19]). Non-commercial applications are also developed in the MIR research field to automatically retrieve gestural information from a combination of audio-video or motion capture recordings of a performance and with the use of predictive models [1].

1.2.2 Seeking information, demonstrations and feedback

Two other tasks the learner wants to perform is to search for information and instructions and to find models to observe. These two tasks might be complemented with the will to receive feedback on the comprehension and on the execution on the instrument of what was learnt. Various sources of information for these tasks include text-based questions and answers on forums or social networks, blog posts and dedicated Websites exploiting various types of media. Nowadays, audio and video recordings production is easily available to anyone. Distribution is also facilitated with the use of general platforms for videos (Youtube [28], DailyMotion [30], Vimeo [29]) and audio (Soundcloud [36]). These recordings can then be linked and included in the previously mentioned forums, blogs, and Websites. Waldron [22–27] made an extensive study of different online communities of practice where members are seen as ”prosumer”, e. i. users that both produce and consume material posted by the community. In these communities, members can, for example, produce instructional audio-video recordings or post advice on techniques they master, and, on the other hand, post their questions or recordings of their performances to request the other members for feedback and comments. The previously mentioned research projects on automatic analysis of performance can also be used for the task of providing automatic feedback to the users. However, to our knowledge, actual commercial products including automatic feedback only use audio analysis for pitch and rhythm detection (Yousician [38], JellyNote [13]).

A special case of applications can include all this information on a single support. Just as a master would annotate the learner score so that he can go home and practice with that supplemental information, digital scores made possible the concept of augmented score. Augmented scores were first introduced in the VEMUS/IMUTUS project [32, 33] and took the form of annotated scores were the annotations could take various forms: graphical, textual, aural and sonic. VEMUS/IMUTUS also introduced the idea of displaying various analysis curves of the performance in parallel to the score (frequency curves, sonograms, envelope curves, etc.). Nowadays, augmented scores can include data from multimodal recordings of performance and from the analysis of these performances.

1.2.3 Play With Others

Another activity that is frequent in private and group lessons is to play along with the master or with peers. This s a rewarding activity and an important motivational factor. It is often something that a learner would miss when he is alone to practice. As we have seen, nowadays most score editors and score sharing platforms offer playback functionalities. These ”virtual players” can use different technologies from MIDI playback to sound synthesis or sampling to synchronization with audio or even audio-video recordings. Some will only play the instrument as others will also include accompaniment by other instruments. Dedicated software like Band-in-a-Box [40] has been long used for that purpose. However, playing along a virtual peer or band is not as rewarding as playing with real peers. For that purpose, projects like I-Maestro [34] introduced the idea of playing over distance. Game platforms like Rocksmith [37] or learning environments like Yousician [38] also use the idea of challenging other users in network performance of a score.

1.2.4 Structure the Learning Path

Finally, as we have seen the availability of online music resources for learning is vast and disparate. Learners often

Tasks	Activities	Technologies	Examples
Search for material	2	digital score editor	Finale [2], Sibelius [3], MuseScore [4], Guitar Pro [5], TuxGuitar [6], PowerTab [7]
		collaborative score editor	JamShake [8], LeadSheetJS [9]
		score / tablature sharing platform	General: NoteFlight [10], Scorio [11], flat.io [12], JellyNote [13], Soundslice [14] Tablature: Songsterr [15], Ultimate Guitar [16]
		automatic audio analysis	Chordify [17], Capo [18], GuitarMaster [19], Songs2See [20]
		automatic multimodal analysis	see [1] for a review
Seek for information and explanation	3	Blog / Forum / Website	Hooktheory [21], specialized Websites (look Waldron [22–27] for references)
		Instructional videos	General purpose video-sharing platforms (Youtube [28], Vimeo [29], DailyMotion [30]), PRAISE [31], TELMI
		augmented score	IMUTUS / VEMUS [32, 33] , I-Maestro [34], PRAISE (LeadSheetJS) [9]
Observe demonstrations	4	Instructional videos	see previous
		3D motion capture and synthesis	I-Maestro [34]
Ask for feedback / evaluation	5	Forum, Website	see previous
		social network	general purpose social networks, PRAISE [31]
		video-sharing platform	General purpose video-sharing platforms, specialized Websites
		audio sharing platform	Music Circle (PRAISE) [35], SoundCloud [36]
		collaborative score editor / augmented score	see previous
		automatic audio analysis	JellyNote [13], Soundslice [14], Rocksmith [37], Yousician [38], IMUTUS / VEMUS [32, 33], I-Maestro [34], PRAISE [31], Songs2See [20], Magna Quest [39]
		automatic multimodal analysis	see previous
		motion capture / sensors	I-Maestro [34]
Play with others	6, 7	Midi / synthesis / sample playback	score editors, score sharing platforms, IMUTUS / VEMUS [32, 33], I-Maestro [34], PRAISE [31]
		accompaniment software	Band-in-a-Box [40]
		audio recording	IMUTUS / VEMUS [32, 33] , I-Maestro [34], PRAISE [31], JellyNote [13]
		serious game	Rocksmith [37]
		play over a distance / online competition	I-Maestro [34], Yousician [38]
Structure his / her learning	1, 2	serious game	Rocksmith [37], Magna Quest [39]
		learning management system (LMS)	Yousician [38] , IMUTUS / VEMUS [32,33] , I-Maestro [34], PRAISE [31], Songs2See [20]

Table 2: Technologies used to assist learning and teaching activities.

have to search among many different sources to find resources that fit their needs. This in itself can be a source of motivation lost. However, there exist platforms that try to solve that problem by providing a complete environment dividing the learning in achievable and significant steps that can quantitatively and qualitatively be evaluated. These platforms are of two kinds: serious game and learning management system (LMS). There is no absolute definition of serious games, but what is generally agreed upon is that they are a category of games that uses entertainment and multimedia to convey an experience to the user [41]. In the case of music education, that experience will take the form of a musical skill. Serious games for music are derived from electronic rhythm games that appeared in the 1990s. They have the specificity of using real instruments as game controllers. Rocksmith [37] from Ubisoft is a well-know example for the guitar. Magna Quest [39] is another example emerging from research and applied to the violin.

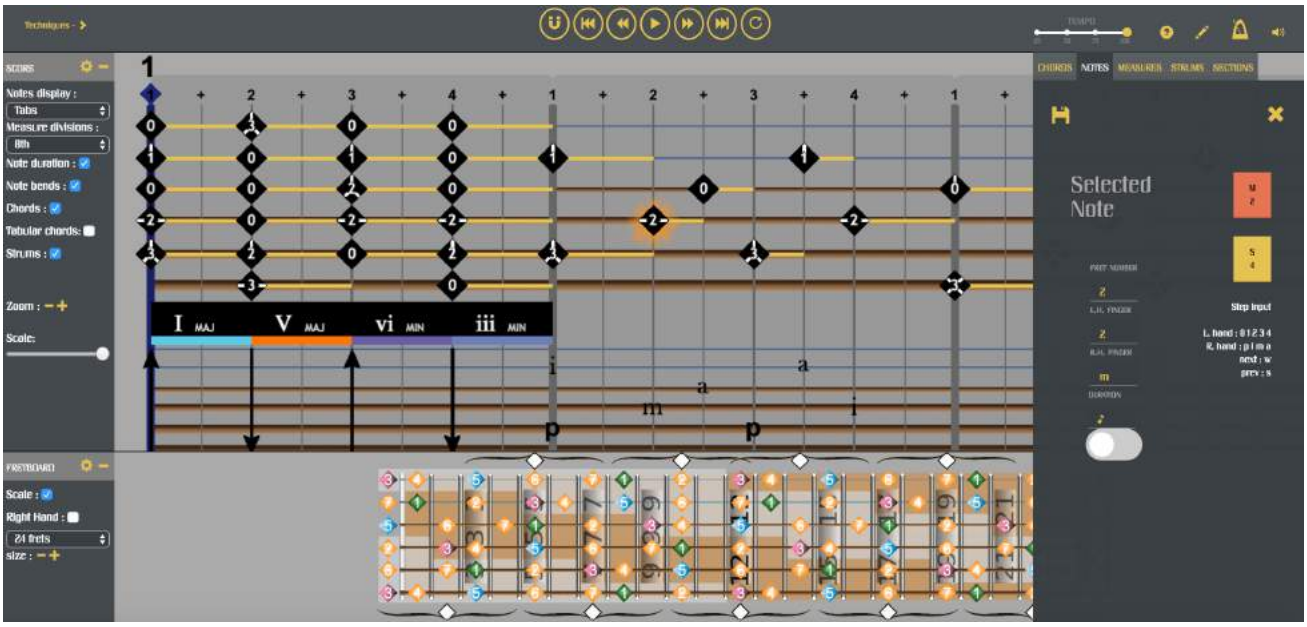
On the counter part, LMS dedicated to instrumental learning or said otherwise computer-assisted music pedagogy for instrumental learning are derived from the digital form of a pedagogy named programmed learning. Actual LMS use game design elements to keep the learner motivation, this is what is called gamification [42].

In LMS, learning can be organized in levels where the learner can progress on different paths. The learners can compare themselves with others throughout the use of leaderboards. They can follow their progress with progression bars and dashboards. They can unlock new learning paths by taking tests and receiving badges. These badges can be shared on social networks. Learning analytics tools can also be used to inform the learners of their progress and recommend them activities to go further. All these tools are at the disposition of pedagogical content developers to help them create motivating learning environments. The research projects we review all include an LMS. I-Maestro [34] also includes the idea of collaborative work. While PRAISE [31] introduce social network tools with the Music Circle where learners and masters can share their performances and annotate other learners performance recordings.

2. THE NOVAXE PROJECT

This project will offer a new gesture rich guitar notation as well as an augmented score, allowing learners to access different dimensions of the performance and to create links between the gestural instructions found in a score and the one captured during real performances using a set of sensors. It implies the creation of analysis tools to extract score and gestural parameters from a performance. These parameters are useful to align the score and the performance, to complement the score with performance instructions, to transcribe a score from a performance and to provide feedback to a learner on his performance of a score. The project aims to explore the hypothesis that interaction and collaboration throughout peer reviews and emulation combined with observational learning from a master but also from self-observation can be beneficial for musical instruments learning.

This project was created to address some of the fundamental problems in modern music education. It consists of three main facets; advanced notation, Web-based pedagogical technology, and a highly structured educational system. The main focus of the project is to enhance the musical education experience by making it more intuitive, rewarding, and efficient for both the casual student and the serious apprentice. The project



Notes display : Tabs
Measure divisions : 8th
Note duration : ☑
Note bends : ☑
Chords : ☑
Tabular chords: □
Strums : ☑
Zoom : − +
Scale:
Scale : ☑
Right Hand : □
24 frets
size : − +

FRET NUMBER 2
L.H. FINGER 2
R.H. FINGER m
DURATION

Step input
L. hand : 0 1 2 3 4
R. hand : p i m a
next : w
prev : s

Figure 1: A Novaxe score in the Web platform player with all editions and properties panels open.

is designed to answer three main challenges currently faced by guitar teachers.

First, the majority of guitar students use various notations: standard notation, standard arranged notation, basic tablature, advanced tablature, lead sheets, and the chord and lyric style. Each of these systems has strengths and weaknesses, often increasing workload, while compromising learning efficiency. The project alternate notation alleviates the core challenges faced by many music students.

Second, social networks and cloud computing has changed the way we socialize, collaborate and learn. Internet services and software are more connected than ever. With the propagation of Web applications, driven by recent technological advances, we can access complex data-driven applications from personal computers as well as from a myriad of smart devices. As we have seen in the first part of this article, music learning and production has not escaped these new paradigms. As Martín e al. notes: ”with the rise of online music communities using performance or pedagogical applications, there is an increasing need for tools for manipulating music scores. There is also a need for Web-based tools for visualizing, playing, and editing [scores] collaboratively [9].”

Finally, learning to play a musical instrument is a long process as musical gesture enters a category of gestures referred to as expert gesture. This type of gesture requires years of deliberate practice and hours of repetitive exercises in order to achieve a high level of proficiency. These high-level skills are commonly transmitted from a master to an apprentice throughout hours of guided observations. However, this teaching method has always faced some issues. The periods of contact between the master and the apprentice are often short and punctual. In between these periods, the apprentice is left all alone for self-learning. In the case of pure online learning, the master-to-student link is even compromised by the lack of physical proximity. Sensors, sound and video analysis are used to mitigate this indirection, and provide the learner with valuable feedback. This helps to maintain the learner motivation and reduces the risk of developing wrong technique that would then take even more practice time to correct.

2.1 The Online Learning Environment

This project is therefore an attempt to answer these challenges faced by music and guitar teaching at the age of interactive and collaborative Web technology. The pedagogy and the innovative musical notation used in the project were developed by Vandendool over more than ten years of personal experience as a private guitar teacher [43, 44]. The pedagogy includes concepts from music theory such as scale degrees, chord tones, harmonic function of chords, rhythm necklace and technical information for both hands. The alternative notation system of the project has the capacity to display this extra information, giving the student a very high level of technical and musical insight. This information is leveraged to create relationships between scores according to the learners skill set, using characteristics such as the progression of chord degrees, fret position, technical information and difficulty.

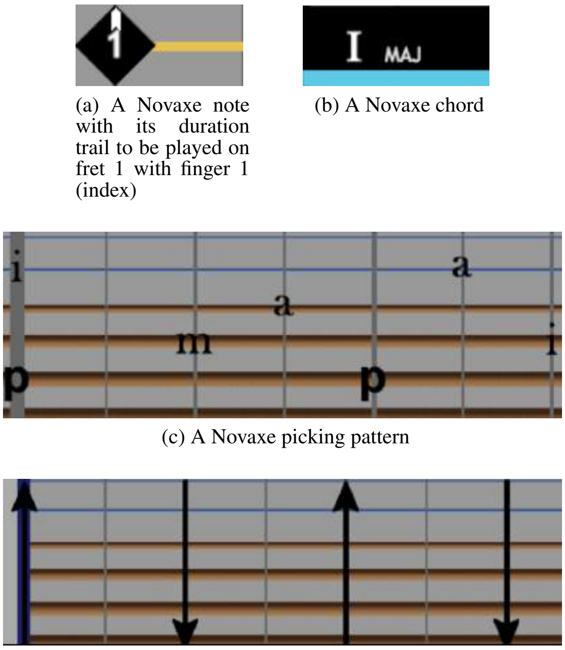
These pedagogical tools are ported to the digital world. The platform in development includes a score editor and player entirely based on the most recent Web technologies (HTML5, CSS3, and several Javascript libraries). It permits to import scores from other digital formats like Guitar Pro [5] and MusicXML [45] and to convert them into the project notation format. It is also possible to create scores from scratch or to complement these with the editor. The editor let the user add technical and theoretical information to the score including right-hand technique, left hand fingering, notes function in a chord and chord function in a chord progression. This information is organized in visual layers that can be displayed or not depending on a student's level. The player presents several useful options to the user, for example the ability to loop sections of the score or to modify the playback speed.

2.2 Augmented Score

The project is currently at the stage of being populated with scores and guitar lesson material. For that purpose, a set of sensors is in development to capture and analyse information directly from the performance. The usage of these sensors responds to three objectives of the project: automated score transcription, real-time feedback, and collection of data related to musical technique. It will also permit the creation of augmented scores. Augmented scores use a combination of the notation system with multimedia data from the performance of the piece. The multimedia data is comprised of sound and sound analysis curves, multi-point-of-view videos and motion capture data. Scores can be played at various speeds and navigated with synchronized data. This permits the user to explore the score in the usual fashion (notes plotted over a time axis), but also using a data-mining dimension, unveiling multiple layers of information about the performance. Similar ideas were explored for musical education in the IMUTUS / VEMUS [46] and iMaestro [47] project with promising results.

A musical score is a symbolic representation giving instructions to a musician for the performance of a musical piece. In this project, information in a score (figure 1 and 2) is displayed on a grid where the horizontal axis is divided into fractions of beats similar to the piano roll representation used in MIDI sequencers. As it is the case with tablature, the vertical axis represents the strings of the guitar and indicates with a diamond which note is played on a given string at a given beat. The number inside the diamond indicates the fret position and the white stripes around the diamond indicate the finger to use. The duration of the notes is indicated with a yellow trail that starts from the beat division where the note should be played and stops at the beat division where it should be released. In that notation, silences are simply represented by the absence of notes. This notation is a transcription of play instructions, therefore, only the notes that are sounded are indicated. Consequently, notes that would be indicated with a slur in the western notation can have their duration spread between measure.

In this representation, every note is a discrete event. The same score can be played at various tempos. A single score can have multiple performances associated with it. These performances will be of different time length and depending on the interpretation, every single event of the continuous time space of the performance will slightly vary from the reference beat at which it should have been played. In order to synchronize the recording of a performance with its symbolic score representation, the multimodal data streams need to be aligned and quantized. This can be done in real time by recording the performance while providing the musician with a reference tempo or by post-processing the recorded performance. In this project, both real time processing and post-processing will be used to respond to different situations. Real-time will be used for immediate display of information to the player. Post-processing will be used for the batch treatment of multiple performances and for deeper analysis of performances.



(a) A Novaxe note with its duration trail to be played on fret 1 with finger 1 (index)

(b) A Novaxe chord

(c) A Novaxe picking pattern

(d) A Novaxe strumming pattern

Figure 2: Novaxe score elements

2.3 Future Work

The aim behind the Novaxe project is to build an environment that will permit the learners to perform all the tasks identified in Table 2 in a single Web application. Our hope is that creating such a cohesive and integrated learning experience will contribute in maintaining the learner's motivation in the critical period in-between lessons or in a self-taught situation. Once the environment is online, our next research questions will be to observe the creation of communities of practice and the interaction between learners and peers, learners and masters, and learners and virtual peers and masters.

The project is developed using the Agile methodology. That means that features are integrated in an incremental manner. That permits every step to be validated by groups of testers. The first iteration of the project is actually available online [1] with a basic set of features. In that alpha version, visitors can access public scores and visualize these in the score player. Users can register and log in to access their private score. Private scores are scores that a user has imported using the Novaxe uploader. Users can share scores with other users of the system. That is the first step of the social network and learning

[1] http://www.novaxe.com

environment (for example, teachers can use that feature to share exercises with their students). The alpha version of the player offers different playback functionalities. The users can modify the tempo and loop throughout sections of the score. The notes to be played can be visualized using the Novaxe notation system or on a fretboard representation. An online editor is already partially developed and will become available in subsequent versions. A search engine based on musical information of the score (ex.: chord progressions, techniques, etc.) is also in development. We believe that such a basic set of features based on an individual guitar master experience was necessary to ”start a discussion” with potential users of the project. The Agile development methodology will subsequently help us to develop a platform that will become closer to the user needs at each iteration.

3. CONCLUSIONS

The ideal master-to-learner musical instrument learning situation involves different modalities from reading a symbolic representation of a musical excerpt to regulating one’s action by visual and aural comparison with an expert musician or with peers. This complex feedback loop is lost in periods in between lessons or in the case of self-taught learning. But technological innovations can be used to regain access to some of these modalities, for example, with sound and video recordings. However, even with recordings, what is still missing is the possibility to interact and receive feedback from a master or from peers. This has been partially alleviated with the use of social networks and automatic feedback tools.

Nowadays, the diversity of online music resources is creating the need for a cohesive and integrated learning experience. We have seen how the computer games industry has influenced the way we approach the development of virtual learning environments in every education field but also in music. We have seen how game elements can be used to maintain the learner motivation even in the case of repetitive exercises. We have seen how these LMS can be coupled with social networks to create communities of practice around a learning platform.

The use of the digital and Web technologies in instrumental learning opens new perspectives. For example, as Waldron notes, the proximity with a master or other players is no longer a constraint to belong to a community of practice of a particular instrument or style [23, 25, 26]. It also relates the symbolic musical notation more closely to the performance with the uses of augmented score giving the user access to more dimensions of a musical excerpt. However, it also raises new questions. It is not yet clear what uses the learner is able to make of all the extra information he is receiving in the form of graphical representations of his performance and how he is able to use the diverse types of feedback. Does that really provide him with different perspectives on the matter to learn and will that teach him about the subject matter but also about himself as a learner [48]? Or does that make him dependent on the learning environment? Is the learner still able to develop his own proprioception or is he always relying on the system? The usage of artificial intelligence agent to take care of the user preferences and learning style might seem appealing, but might it be at the risk of limiting the learner to what is naturally attractive to him? Answering these questions is probably a required step to develop the necessary critical thought to give these technologies their proper place in the musical learning tools arsenal.

Acknowledgments

This research is funded by a Mitacs acceleration grant in the context of the Novaxe gesture toolkit project[2]. The authors would like to thank all the development team: Mark, Andrew, Christelle, Mathieu and Etienne and the industrial partner, Orval Vandendool.

4. REFERENCES

[1] A. Perez-Carrillo, J.-L. Arcos, and M. Wanderley, “Estimation of guitar fingering and plucking controls based on multimodal analysis of motion, audio and musical score,” in *Proceedings of the 11th International Symposium on CMMR*, Plymouth, United-Kingdom, June 16-19 2015.

[2] Finale. [Online]. Available: http://www.finalemusic.com

[3] Sibelius. [Online]. Available: http://www.avid.com/sibelius

[4] Musescore. [Online]. Available: https://musescore.org

[5] Guitar pro. [Online]. Available: http://www.guitar-pro.com/

[6] Tuxguitar. [Online]. Available: http://tuxguitar.herac.com.ar

[7] Powertab. [Online]. Available: http://www.power-tab.net

[8] Jamshake. [Online]. Available: https://www.jamshake.com

[9] D. Martín, T. Neullas, and F. Pachet, “Leadsheetjs: A javascript library for online lead sheet editing,” in *TENOR 2015 First International Conference on Technologies for Music Notation and Representation*, Paris, France, 28-30 may 2015 2015.

[10] Noteflight. [Online]. Available: https://www.noteflight.com

[11] Scorio. [Online]. Available: http://www.scorio.com

[12] Flat.io. [Online]. Available: https://flat.io

[13] Jellynote. [Online]. Available: https://www.jellynote.com

[14] Soundslice. [Online]. Available: https://www.soundslice.com

[2] https://www.mitacs.ca/en/projects/novaxe-gesture-toolkit

[15] Songsterr. [Online]. Available: https://www.songsterr.com

[16] Ultimate guitar. [Online]. Available: https://www.ultimate-guitar.com

[17] Chordify. [Online]. Available: https://chordify.net

[18] Capo. [Online]. Available: http://supermegaultragroovy.com/products/capo/

[19] Guitarmaster. [Online]. Available: http://www.guitarmaster.co.uk

[20] C. Dittmar, E. Cano, J. Abeßer, and S. Grollmisch, "Music information retrieval meets music education," in *Multimodal Music Processing. Dagstuhl Follow-Ups*, M. Müller, M. Goto, and M. Schedl, Eds. Germany: Dagstuhl Publishing, 2012, vol. 3, ch. 6, pp. 95–120.

[21] Hooktheory. [Online]. Available: https://www.hooktheory.com

[22] J. Waldron and K. K. Veblen, "The medium is the message: Cyberspace, community, and music learning in the irish traditional music virtual community," *Journal of Music, Technology and Education*, vol. 1, no. 2–3, pp. 99–111, November 2008.

[23] J. Waldron, "Exploring a virtual music community of practice: Informal music learning on the internet," *Journal of Music, Technology and Education*, vol. 2, no. 23, pp. 97–112, November 2009.

[24] ——, "Locating narratives in postmodern spaces: A cyber ethnographic field study of informal music learning in online community," *Action, Criticism & Theory for Music Education*, vol. 10, no. 2, December 2011.

[25] ——, "Conceptual frameworks, theoretical models and the role of youtube: Investigating informal music learning and teaching in online music community," *Journal of Music Education and Technology*, vol. 4, no. 2–3, pp. 189–200, 2011.

[26] ——, "Youtube, fanvids, forums, vlogs and blogs: Informal music learning in a convergent on - and offline music community," *International journal of music education*, pp. 1–16, February 2013.

[27] ——, "User-generated content, youtube and participatory culture on the web: Music teaching and learning in two contrasting online communities," *Music Education Research*, vol. 15, no. 3, pp. 257–274, April 2013.

[28] Youtube. [Online]. Available: https://www.youtube.com

[29] Vimeo. [Online]. Available: https://vimeo.com

[30] Dailymotion. [Online]. Available: http://www.dailymotion.com

[31] Practice and performance analysis inspiring social education. [Online]. Available: http://www.iiia.csic.es/praise/

[32] D. Fober, S. Letz, and Y. Orlarey, "Vemus - feedback and groupware technologies for music instrument learning," in *Proceedings of the 4th Sound and Music Computing Conference*, Lefkada, Greece, 11-13 July 2007, pp. 117–123.

[33] S. Raptis, A. Askenfelt, D. Fober, A. Chalamandaris, E. Schoonderwaldt, S. Letz, A. Baxevanis, K. F. Hansen, and Y. Orlarey, "IMUTUS - An Effective Practicing Environment For Music Tuition," in *Proc. of the International Computer Music Conference*, 2005, pp. 383–386.

[34] K. Ng, B. Ong, T. Weyde, and K. Neubarth, "Interactive multimedia technology-enhanced learning for music with i-maestro," in *World Confenrence on Educational Multimedia, Hypermedia and Telecommunications*, June 2008.

[35] Music circle. [Online]. Available: http://goldsmiths.musiccircleproject.com

[36] Soundcloud. [Online]. Available: https://soundcloud.com

[37] Rocksmith. [Online]. Available: https://rocksmith.ubisoft.com

[38] Yousician. [Online]. Available: http://yousician.com

[39] F. Dubé, J. Kiss, and H. Bouldoire, "Gamification and learner engagement: A "learning the violon" implementation interface example," in *In proceedings of International Symposium Learning and Teaching Music in the Twenty-First Century: The Contribution of Science and Technology LTM21/AEM21*, Montreal, Quebec, Canada, 2015.

[40] Band-in-a-box. [Online]. Available: http://www.pgmusic.com

[41] F. Laamarti, M. Eid, and A. El Saddik, "An overview of serious games," *International Journal of Computer Games Technology*, p. 15, October 2014.

[42] S. Deterding, D. Dixon, R. Khaled, and L. Nacke, "From game design elements to gamefulness: Defining "gamification"," in *Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments*. Tampere, Finland: ACM, New York, NY, USA, September 28-30 2011, pp. 9–15.

[43] M. Vandendool, "Musical notation systems for guitar fretboard, visual displays thereof, and uses thereof," US Patent App. 13/740,842, July 18 2013. [Online]. Available: http://www.google.ch/patents/US20130180383

[44] ——, "Musical notation systems for guitar fretboard, visual displays thereof, and uses thereof," CA Patent App. CA 2,802,201, July 12 2013.

[45] Music xml. [Online]. Available: http://www.musicxml.com/

[46] D. Fober, J. Bresson, P. Couprie, and Y. Geslin, "Les nouveaux espaces de la notation musicale," in *Actes des Journées d'Informatique Musicale JIM2015*, Montreal, Quebec, Canada, 7-9 May 2015.

[47] B. Ong, A. Khan, K. Ng, P. Bellini, N. Mitolo, and N. Paolo, "Cooperative multimedia environnements for technology-enhanced music playing and learning with 3d posture and gesture supports," in *Proceedings of the International Computer Music Conference (ICMC 2006)*, Tulane University, USA, 6-11 November 2006, pp. 135–138.

[48] T. W. Malone, "Toward a theory of intrinsically motivating instruction," *Cognitive science*, vol. 4, pp. 333–369, 1981.

KEY INFERENCE FROM IRISH TRADITIONAL MUSIC SCORES AND RECORDINGS

Pierre Beauguitte
Dublin Institute of Technology
pierre.beauguitte@mydit.ie

Bryan Duggan
Dublin Institute of Technology
bryan.duggan@dit.ie

John D. Kelleher
Dublin Institute of Technology
john.d.kelleher@dit.ie

ABSTRACT

The aim of this paper is to present techniques and results for identifying the key of Irish traditional music melodies, or *tunes*. Several corpora are used, consisting of both symbolic and audio representations. Monophonic and heterophonic recordings are present in the audio datasets. Some particularities of Irish traditional music are discussed, notably its modal nature. New key-profiles are defined, that are better suited to Irish music.

1. INTRODUCTION

Key detection is a common task in Music Information Retrieval (MIR), and has been a part of the Music Information Retrieval Evaluation eXchange (MIREX) since its start in 2005. Motivations for it include automatic analysis and annotations of large databases. The present study focuses on key identification for Irish traditional music tunes. Key-finding algorithms are tested on two collections of audio recording, of session and solo recordings, representing 636 tunes overall. Symbolic transcriptions have been compiled for all the tunes. A range of methods from the literature are benchmarked on both the audio and symbolic data. Some modifications to these methods, as well as new methods, including a set of parametric models, are presented and tested.

A musical key consists of a tonic note, represented by a pitch class (C, $C\#$, D...), and a mode, or rather mode family, which can be minor or major. Consequently, there are always 24 candidate keys, for the 12 semitones of the octave and the two considered modes. Enharmonic equivalence is used, which means that we do not distinguish between different spellings of the same note in the twelve-tone equal temperament, *e.g.* $D\#$ and Eb. Throughout the paper we will adopt the convention of denoting major keys by upper-case letters, and minor keys as lower-case ones.

The standard approach to identifying keys in a musical piece is to use key-profiles [1]. They can be seen as vectors assigning weights to the twelve semitones, denoted $(p[i])_{i=0,\dots,11}$. One key-profile per mode is defined for the tonic note C, and transposition to the another tonic note is performed by rotating the elements in the vector. A histogram $(h[i])_{i=0,\dots,11}$ of cumulative durations of each pitch class in the musical excerpt is generated, and the score is the weighted sum of the histogram with the key-profile.

$$s(p,h) = \sum_{i=0}^{11} p[i] * h[i] \quad (1)$$

The estimated key is then the one corresponding to the highest scoring profile:

$$key(h) = key(\arg\max_{p \in \mathcal{P}} s(p,h)) \quad (2)$$

where $\mathcal{P}$ is the set of 24 key-profiles representing candidate keys.

This method only needs a pitch class histogram from the musical excerpt. From a symbolic representation, obtaining this is straightforward. From an audio representation, an extra step of computing a chromagram is required, which does not represent any significant difficulties. It can however add some noise to the histogram because of the harmonics present in an audio signal. The resulting pitch class histogram is, in the classification proposed in [2], a low-level global descriptor.

Other methods for key-identification are based on higher-level features. For example, in [3] the intervals of a melody are analyzed, which presupposes that an automatic transcription of the signal has been performed beforehand. In [4], an HMM is trained to estimate the key from a sequence of chords.

In preparing this paper we also experimented with some machine learning (ML) models for key-detection (such as multinomical regression models). Generally, ML models perform best on relatively balanced datsets; consequently, in order to train our ML models we introduced transposed tunes into the dataset in order to balance the distribution in the data. However, the performance of these ML models was relatively weak and as a result we do not include a description of them nor the data preparation carried out for them in this paper.

The main contribution of this paper is a set of new key-profiles, outperforming existing keys-profiles on the corpora considered. In Section 2, we present the key-finding method, and existing key-profiles defined in the literature. In Section 3, new key-profiles are introduced. Section 4 gives a description of the datasets. Section 5 gives the details and results of the experiment using the profiles. Section 6 presents a parametric extension of one of the models, the method used to select the parameters, and the performance of this model. Finally Section 7 discusses the study and indicates future ideas for research.

2. RELATED WORK

2.1 Triads

Certainly the most naive way to define a key-profile is to consider only the triad of the tonic chord. For example, in C it is expected that the pitch classes of the tonic C, the third E and the fifth G will be the most frequent, as reflected in the key-profiles:

$$\begin{aligned} p_{\text{triad}}(C) &= [1,0,0,0,1,0,0,1,0,0,0,0] \\ p_{\text{triad}}(c) &= [1,0,0,1,0,0,0,1,0,0,0,0] \end{aligned}$$

2.2 Krumhansl-Kessler

The key-profiles established in [5] were obtained by perceptual experiments, in contrast with the triads presented above which were motivated by musical theory. Subjects were asked to rate how well the different pitch classes fit with a short musical excerpt establishing a key. The Krumhansl-Kessler key-profiles are a well known method for key detection.

$$\begin{aligned} p_{\text{KK}}(C) = [&6.35, 2.23, 3.48, 2.33, 4.38, 4.09, \\ &2.52, 5.19, 2.39, 3.66, 2.29, 2.88] \\ p_{\text{KK}}(c) = [&6.33, 2.68, 3.52, 5.38, 2.60, 3.53, \\ &2.54, 4.75, 3.98, 2.69, 3.34, 3.17] \end{aligned}$$

2.3 Lerdahl's Basic Spaces

The "basic spaces" defined by Lerdahl in [6] are derived from the diatonic scale of each key. Different weights are given to the degrees of the scale: 5 for the tonic (index 0 for C and c), 4 for the fifth (index 7 for C and c), 3 for the third (index 4 for C, 3 for c), 2 to the rest of the diatonic scale, and 1 to the remaining semitones.

$$\begin{aligned} p_{\text{Lerdahl}}(C) &= [5,1,2,1,3,2,1,4,1,2,1,2] \\ p_{\text{Lerdahl}}(c) &= [5,1,2,3,1,2,1,4,2,1,2,1] \end{aligned}$$

It is worth noting that the natural minor scale, or Aeolian scale, is considered here: the natural seventh (index 10) is taken as part of the scale, not the augmented seventh (index 11) as would be the case with the harmonic minor scale.

2.4 Leman's Tone Center Images

In [7], the *simple residue image* (or R-image) of a chord is generated as a weighted combination of the undertone series of the tonic. The *tone center images* are then derived by summing the R-images of the chords present in the common cadences. The three typical cadences selected in [7] are

$$\left\{ \begin{array}{llll} \text{I} & \text{IV} & \text{V} & \text{I} \\ \text{I} & \text{II} & \text{V} & \text{I} \\ \text{I} & \text{VI} & \text{V} & \text{I} \end{array} \right.$$

where the type of the chord the depends on the scale considered. In the major case, the classic major scale (Ionian) is used. However in the minor case, the harmonic scale is chosen, where the seventh degree is one semitone higher than in the natural scale.

The Tone Center Images (TCI) are then obtained by summing the R-images of the chords, weighted by how often they occur in the cadences, and normalizing:

$$6 * \text{I} + 3 * \text{V} + \text{II} + \text{IV} + \text{VI}$$

After normalization, the key-profiles obtained are:

$$\begin{aligned} p_{\text{Leman}}(C) = [&0.36, 0.05, 0.21, 0.08, 0.24, 0.21, \\ &0.05, 0.31, 0.07, 0.24, 0.09, 0.10] \\ p_{\text{Leman}}(c) = [&0.34, 0.11, 0.15, 0.25, 0.11, 0.25, \\ &0.02, 0.31, 0.24, 0.09, 0.12, 0.14] \end{aligned}$$

3. NEW PROFILES

In this section two new pairs of key-profile are introduced.

3.1 Basic spaces for modal scales

The key-profiles introduced in 2.3 are based on the natural scales, or Ionian mode for major and Aeolian mode for minor. In Irish music, two other modes are commonly used: the Mixolydian mode (major with a minor seventh) and the Dorian mode (minor with a major sixth). The two basic spaces are modified so that they are suited to both major modes (Ionian and Mixolydian) and minor modes (Aeolian and Dorian). This is done by setting $p[10] = p[11]$ in the major case, and $p[9] = p[8]$ in the minor case:

$$\begin{aligned} p_{\text{Lerdahl}^\star}(C) &= [5,1,2,1,3,2,1,4,1,2,2,2] \\ p_{\text{Lerdahl}^\star}(c) &= [5,1,2,3,1,2,1,4,2,2,2,1] \end{aligned}$$

This idea of considering the minor and major seventh as equivalent has already been used for the task of Irish traditional music tune transcription and recognition in [8].

3.2 Cadences

These key-profiles are inspired by Leman's tone center images. As mentioned in 2.4, cadences play an important role in establishing a tonal center. In Irish traditional music, the most commmon cadence is I - IV - V - I [9]. In the case of minor tunes, we also consider the chord sequence VII - VII - I - I, often used in accompaniments. Consequently the formulae to obtain the key-profiles are

$$\left\{ \begin{array}{ll} \text{Major:} & 2 * \text{I} + \text{IV} + \text{V} \\ \text{Minor:} & 4 * \text{I} + 2 * \text{VII} + \text{IV} + \text{V} \end{array} \right.$$

Instead of considering R-images, chords are simply represented by their triad, as introduced in 2.1. The resulting key-profiles are:

$$\begin{aligned} p_{\text{Cadences}}(C) &= [3,0,1,0,2,1,0,3,0,1,0,1] \\ p_{\text{Cadences}}(c) &= [5,0,3,4,0,3,0,5,1,0,3,0] \end{aligned}$$

Finally, these profiles are also modified to account for the Mixolydian and Dorian modes:

$$p_{\text{Cadences}^\star}(C) = [3, 0, 1, 0, 2, 1, 0, 3, 0, 1, 1, 1]$$
$$p_{\text{Cadences}^\star}(c) = [5, 0, 3, 4, 0, 3, 0, 5, 1, 1, 3, 0]$$

4. DATASETS

This section introduces the datasets used for this study.

4.1 Audio datasets

Two sets of recordings are used in this study, representing overall 636 audio items. In both cases, each tune was annotated with key information by the first author.

4.1.1 Foinn Seisiún

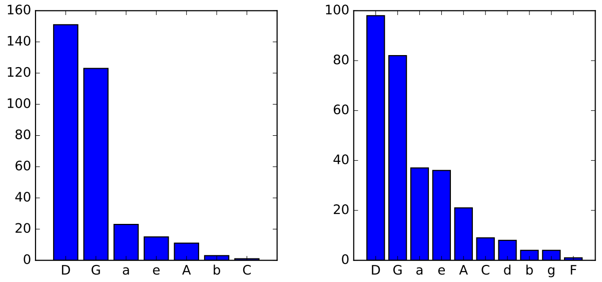
This collection consists of session recordings accompanying the Foinn Seisiún books published by the Comhaltas Ceoltóirí Éireann organisation. Instruments in the recordings include flute, tin whistle, uillean pipes (Irish bagpipes), accordion, concertina, banjo, piano, guitar, bodhran (drum). They offer good quality, homogeneous examples of the heterophony inherent to an Irish traditional music session. The whole collection consists of 3 CDs, representing 327 tunes. The first 2 CDs (273 tunes) are available under a Creative Commons Licence, while the third is commercially available. In five instances, two recordings of a same tune are present. In four cases, we decide to keep both as different items in our dataset, since the set of instruments recorded is different. Only in one case is the exact same recording present, in which case we discard one of the recordings. In the end, this dataset contains 326 distinct recordings, and is denoted *FS_{audio}* in the rest of the article.

4.1.2 Grey Larsen's 300 gems

Grey Larsen's recordings, accompanying the book *300 Gems of Irish Music for All Instruments*, is a set of MP3 files commercially available. They consist of studio quality recordings of tunes played on Irish flute, tin and low whistles, and anglo concertina. None of the 300 audio recordings are of the same tune. This dataset is denoted GL_{audio} in the rest of the article.

4.2 Symbolic datasets

For each tune in both audio corpora, a symbolic transcription was collected in ABC format. The majority of the transcriptions were found online, mostly on the collaborative website `www.thesession.org`. A small number of tunes were not available, in whose cases the audio recordings were manually transcribed to ABC by the first author. It is important to note that the symbolic transcriptions do not correspond exactly to the music played in the audio recording. Indeed Irish music is always interpreted with ornaments and small variations. The score is rather seen as an outline of the melody to be played. The difference between recordings and scores is even more clear for the session recordings: the audio signal is then heterophonic, as the different musicians are not playing exactly the same melody.



(a) Keys in *FS* (b) Keys in *GL*

Figure 1. Distributions of keys in *FS* and *GL*

This time, all redundant copies of duplicate tunes present in the Foinn Seisiún collection are discarded, as in such cases the score remains the same even though the recording differs. Hence FS_{symb} contains 322 items, and GL_{symb} 300.

4.3 Distribution of keys

Both datasets are unbalanced in terms of key distribution, as can be seen on Figure 1. These distributions are actually quite representative of the reality of how Irish music is played. The keys of D and G are indeed the most common in sessions, in part due to the fact that some instruments are limited to these scales (*e.g.* keyless flute, whistle, uileann pipes, . . .).

5. EXPERIMENT 1

5.1 Pitch class histograms extraction

The first step of the algorithm is to extract a pitch class histogram from the musical piece. In the case of symbolic representation, this poses no difficulty. The software `abc2midi` [1] was used to parse the ABC files.

For each audio recording, a chromagram was first generated, then the chroma vectors were summed over time. Chromagrams were obtained using the `madmom` [2] library. Several methods of computing the chromas were tested: standard pitch class profile, harmonic pitch class profile [2], and Deep Chroma extractor [10]. This last method, using a deep neural network trained to extract chromas from a spectrogram, consistently outperformed the others. Consequently all the results reported below are obtained with the Deep Chroma method.

5.2 Key search

For each of the key-profile sets, the first step is to normalize the profile:

$$\overline{p}[i] = \frac{p[i]}{\sum_{i=0}^{11} p[i]}$$

This is important for cases where the major and minor profiles do not have the same sum, *e.g.* Krumhansl-Kessler profiles, or the ones based on cadences.

[1] `http://abc.sourceforge.net/abcMIDI/`
[2] `http://madmom.readthedocs.io`

	FS_{audio}	FS_{symb}	GL_{audio}	GL_{symb}
Triad	0.873	0.795	0.671	0.709
KK	0.854	0.869	0.665	0.696
Lerdahl	0.887	**0.890**	0.689	0.777
Leman	0.848	0.829	0.646	0.666
Lerdahl*	0.893	**0.890**	0.711	**0.798**
Cadences	0.883	0.874	0.677	0.770
Cadences*	**0.902**	0.818	**0.713**	0.750

Table 1: MIREX scores for the 4 corpora using different key-profiles

For a musical piece having the pitch class histogram h, the estimated key is then obtained by:

- for each of the 24 normalized candidate key-profiles $\overline{p}$, compute the score $s(\overline{p}, h)$ (Equation (1))
- choose the key whose profile yields the highest score (Equation (2))

The key finding algorithm is unbiased, in the sense that the 24 key-profiles are evaluated in the same way, with no preference for the common keys. Hence the fact that the keys are unbalanced in the datasets as shown in Figure 1 is not an issue for this study.

5.3 Evaluation metrics

The MIREX evaluation metrics is defined as follows: let k be the ground truth annotation, and $\hat{k}$ the estimated key, then the accuracy score for this item is:

$$acc = \begin{cases} 1 & \text{if } k = \hat{k} \\ 0.5 & \text{if } \hat{k} \text{ is the perfect fifth of } k \\ 0.3 & \text{if } \hat{k} \text{ is the relative of } k \\ 0.2 & \text{if } \hat{k} \text{ is the parallel of } k \\ 0 & \text{otherwise} \end{cases}$$

These scores are then averaged across the whole dataset.

5.4 Results

Results are given for the seven pairs of key-profiles considered. All the MIREX accuracy scores are reported in Table 1. The $\star$ superscript indicates the modal versions of the key-profiles presented in Section 3.

Two observations can be made from this table. First, comparing the MIREX scores on the two symbolic datasets shows that inferring the key of the tunes in *GL* is harder than in *FS*. Second, on the *FS* collection, most key-profiles yield better results on the audio data than on the symbolic data. The opposite is true for *GL*. Hence it appears that inferring keys from heterophonic or polyphonic audio is easier than on monophonic recordings. An explanation for this is that the harmonic content is richer in heterophonic and polyphonic signals.

The new key-profiles introduced in Section 3 (Lerdahl*, Cadences and Cadences*) outperform the existing key-profiles on all four datasets. Inspired by Lerdahl's original key-profiles which assign different weights to degrees of the scale (see Section 2.3) we believe that the performance of the Cadences key-profiles can also be improved by introducing weights. Specifically by assigning different weights to the tonic, third and fifth degrees in the triads uses to build the Cadence profiles presented in Section 3.2. To test this hypothesis we ran a second experiment which is described in Section 6.

6. EXPERIMENT 2

6.1 Weighted cadences

The model proposed here is a parameterized version of the previously introduced Cadences profiles. The parameters considered are the three weights given to the three notes of the triads, denoted $W = (w_1, w_3, w_5)$ for the tonic, third and fifth respectively. Then, the following profiles can be derived from the cadences chosen in Section 3.2:

$$\begin{aligned} p_{\text{Cadences}(W)}(C) &= [2w_1 + w_5, 0, w_5, 0, 2w_3, w_1, 0, \\ &\quad w_1 + 2w_5, 0, w_3, 0, w_3] \\ p_{\text{Cadences}(W)}(c) &= [4w_1 + w_5, 0, 2w_3 + w_5, 4w_3, \\ &\quad 0, w_1 + 2w_5, 0, w_1 + 4w_5, \\ &\quad w_3, 0, 2w_1 + w_3, 0] \end{aligned}$$

The modal versions of these profiles, Cadences*(W) are obtained in the same manner as in Section 3.

6.2 Experimental Methodology

Generally, the goal of a model evaluation experiment on a dataset is to estimate the performance of the model on unseen data (i.e, data that was not used to train the model). To achieve this it is traditional in machine learning to first separate a dataset into a training set and test set. The training set is then used to train and compare models in order to choose a single best model and the test set is solely used to evaluate the best model as judged based on the relative performance of models on the training set. The reason for this is that if the performance of models on the test set is used to select the best model then the test set is actually part of training the model. In other words the same data cannot be used to select the best model and to evaluate its performance. In machine learning, using the performance of a model on the test set to select the best model is known as "peeking" at the test set. It is equivalent to allowing the model to look at the test set prior to running the test which is problematic because it can result in optimistic performance scores.

Returning to the concerns of the current paper each set of weights applied to the weighted cadences key-profiles defines a separate key-detection model. Using a grid-search process we can iterate across a grid of model parameters with each point on the grid defining a separate set of weights (and hence a distinct model). The grid-search process provides a mechanism where we can iterate through a set of models and test each model in turn. The problem with this methodology, however, is that if we select the best set of weights (or model) by iterating across the grid and simply selecting the set of weights with the highest performance on the datasets this model selection methodology suffers from the problem of "peeking" introduced in the preceding paragraph. In other words we will find a single best-set of weights on the dataset but the performance of this model on the dataset will not be a realistic measure of the model on unseen data. We will refer to this methodology as the *Best-Weights* method.

An alternative methodology is to use a process called 10-fold cross validation, following the methodology of [11]. The focus of a 10-fold cross-validation process is to estimate the average performance of the models generated by a machine learning algorithm and hyper-parameter set [3] on unseen data. In a 10-fold cross-validation a dataset is split into 10 equally size subsets, or *folds*. Then 10 experiments are run (one per fold). In each experiment 9/10s of the data is used to train a model and the remaining 1/10 (the fold) is used to evaluate the model performance. The overall performance of algorithm and hyper-parameters is then calculated by aggregating the confusion matrices generated by each of the 10 experiments and calculating a performance metric on the resulting matrix. The advantage of a cross-validation methodology is that in each of the 10 experiments distinct data is used to train and test the model. So the final overall accuracy score is representative of the likely performance of a model trained with the tested algorithm on unseen data.

In our context training a model involves selecting the set of weights that perform best on a dataset. So, to utilize a cross-validation methodology to evaluate the weighted cadences approach we need to run a grid-search process in each of the 10 experiments [4] and select the best set of weights for that experiment based on the performance on the 9/10s training portion of the data and then evaluate the performance of these best weights on the (unseen) remaining 1/10 of the data. The advantage of the cross validation methodology is that it provides us with an estimate of the likely performance of a weighted cadences key-profiles on unseen data. This is because in each of the 10 weight tuning and model evaluation experiments (one experiment per fold) distinct sets of tunes are used for weight tuning and evaluation. Consequently, the accuracy scores returned from this process reflect the accuracies of models on unseen tunes (*i.e.* tunes that were not seen during weight tuning). The drawback of this approach, however, is that the weight tuning process in each of these experiments may return different sets of weights. So, although the overall accuracy scores provides an indication of likely performance of a set of weighted cadences key-profiles on unseen data it does not provide an accuracy measure for cadences key-profiles with a specific (fixed) set of weights.

As a result, we decided to use apply both methodologies to the weighted cadences key-profiles. First we applied a cross-validation process to estimate the performance of weighted cadences key-profiles on unseen data. This first step also serves to determine the best grid size to use. We then applied the *Best-Weights* method introduced above, with the chosen grid size, to find the best single set of weights on our dataset.

	FS_{audio}	FS_{symb}	GL_{audio}	GL_{symb}
Cadences(W)	0.891	0.879	0.702	0.769
Cadences$^\star(W)$	**0.908**	0.840	**0.720**	0.745

Table 2: MIREX scores computed from the aggregate matrices after cross-validation on the 4 corpora

6.3 Results

The only hyper-parameter in this experiment is g, the width of the grid. A wide range allows a better fit on the training data, but poses a risk of overfitting it, resulting in poor performance on the test sets. The experiment was performed for g ranging from 2 to 10. The grid size $g = 3$, allowing the weights w_i to take values in $[1, 2, 3]$, gave the best results, and is used for the following results.

Scores calculated from the aggregate matrices after the cross validation on each of the four datasets are presented in Table 2. The models Cadences$^\star(W)$ outperform all other methods on the two audio corpora. However, the Lerdahl* key-profiles evaluated in Experiment 1 remain the best performing ones on the symbolic data. Consequently the rest of this section focuses on Cadences$^\star(W)$ on the audio datasets.

The result of the cross-validation method suggests that the models Cadences$^\star(W)$ generalize well to unseen audio data. In order to obtain one single model (as opposed to the multiple ones resulting from the 10 folds), the *Best-Weights* method was then performed on the combined dataset $(FS + GL)_{audio}$. Grouping the two collections of audio recordings means that the profiles should perform well on both heterophonic and monophonic recordings. The weights obtained are $(3, 1, 2)$, corresponding to the intuition that the tonic and fifth are more important than the third, as in Section 2.3. The resulting key-profiles are:

$$p_{\text{Cadences}^\star(3,1,2)}(C) = [8, 0, 2, 0, 2, 3, 0, 7, 0, 1, 1, 1]$$
$$p_{\text{Cadences}^\star(3,1,2)}(c) = [14, 0, 4, 4, 0, 7, 0, 11, 1, 1, 7, 0]$$

With these profiles, the MIREX scores are 0.901 on FS_{audio} and 0.730 on GL_{audio}, to be compared to the scores in Table 1. The lower score on FS is not unexpected: the grid search maximizes the overall score accross the combined audio collection, regardless of the scores on the individual collections. The overall MIREX score on the combined collection is 0.819, compared to 0.811 with the non-parametric Cadences* profiles.

The confusion matrices for these new key-profiles on the audio collections, and for the Lerdahl* ones on the symbolic datasets (on which they are still the highest scoring ones), are given in Tables 3 to 6. Rows indicate the actual keys in the ground truth annotations, while columns indicate estimated keys. Keys that never occur in either the ground truth or the estimations are omitted.

[3] Hyper-parameters are parameters on a machine learning algorithms as distinct to parameters on a model.

[4] As opposed to the single grid search process that would be used in the *Best-Weights* method.

	D	G	a	e	A	b	C	d
D	149	1	0	0	0	0	0	0
G	4	119	0	0	0	0	0	0
a	5	9	7	0	0	0	1	1
e	8	3	1	3	0	0	0	0
A	2	0	0	0	9	0	0	0
b	2	0	0	0	0	1	0	0
C	0	0	0	0	0	0	1	0

Correct	289
Fifth	6
Relative	6
Parallel	0
Neighbour	17
Other	8

Table 3: Confusion matrix for FS_{audio} with key-profiles Cadences*(3, 1, 2)

	D	G	a	e	A	b	C	f#	B
D	131	3	1	0	0	14	0	1	0
G	0	113	1	4	0	1	0	0	0
a	0	7	11	1	4	0	0	0	0
e	1	1	0	11	0	2	0	0	0
A	0	0	0	0	11	0	0	0	0
b	0	0	0	0	0	2	0	0	1
C	0	0	0	0	0	0	1	0	0

Correct	280
Fifth	0
Relative	19
Parallel	5
Neighbour	9
Other	9

Table 4: Confusion matrix for FS_{symb} with key-profiles Lerdahl*

	D	G	a	e	A	C	d	b	g	F
D	82	4	6	1	1	0	4	0	0	0
G	6	71	4	1	0	0	0	0	0	0
a	5	9	17	0	1	3	2	0	0	0
e	10	6	5	13	2	0	0	0	0	0
A	7	0	2	0	11	0	1	0	0	0
C	0	1	2	0	0	5	1	0	0	0
d	1	0	2	0	0	2	3	0	0	0
b	3	0	0	1	0	0	0	0	0	0
g	0	0	0	0	0	2	0	0	2	0
F	0	0	0	0	0	0	0	0	0	1

Correct	205
Fifth	16
Relative	15
Parallel	8
Neighbour	27
Other	29

Table 5: Confusion matrix for GL_{audio} with key-profiles Cadences*(3, 1, 2)

	D	G	a	e	A	C	d	b	g	F	f#
D	85	1	5	1	0	0	0	6	0	0	0
G	2	71	1	7	0	0	0	1	0	0	0
a	1	13	18	3	1	1	0	0	0	0	0
e	3	1	0	26	1	0	0	5	0	0	0
A	4	0	1	0	14	0	0	0	0	0	2
C	0	1	4	0	0	4	0	0	0	0	0
d	0	0	1	0	0	1	6	0	0	0	0
b	0	0	0	0	0	0	0	4	0	0	0
g	0	0	0	0	0	0	0	0	3	1	0
F	0	0	0	0	0	0	1	0	0	0	0

Correct	231
Fifth	3
Relative	22
Parallel	2
Neighbour	20
Other	22

Table 6: Confusion matrix for GL_{symb} with key-profiles Lerdahl*

The three types of errors taken into account in the MIREX evaluation metrics are highlighted in different shades of blue. Another error occurs frequently in this experiment, named "neighbour". The "neighbour" relationship is defined as follows:

> two keys are neighbours if one is major, the other minor, and the minor one has a tonic one tone above the tonic of the major one.

To take a concrete example, D and e are neighbour keys. In terms of modes, the scales of D Mixolydian and of e Aeolian contain the exact same pitch classes. It is not rare in Irish music that a tune labelled as e minor changes its "tonic center" for a few bars to the neighbour key D, *e.g.* the well known Cooley's reel. On both audio datasets, this type of error is the most common. As such, and although the MIREX evaluation metrics does not take these errors into account, reporting them is relevant.

Relative keys are the next most common errors, on both audio and symbolic datasets. The scales of two relative keys contain, as is the case with neighbour keys, the same pitch classes, if one considers the Aeolian mode. Changes of tonic center in a tune between its key and the relative key are also quite common in Irish music. The high frequencies of these two types of errors can be explained by the specific characteristics of Irish traditional music.

Table 7 gives the percentages of correctly inferred keys per mode. A clear difference in performance appears between the major and minor keys, suggesting that minor keys are harder to detect than major keys.

	FS_{audio}	FS_{symb}	GL_{audio}	GL_{symb}
Major	97.5%	91.1%	80.6%	82.5%
Minor	26.8%	58.5%	39.3%	64.0%

Table 7: Proportions of correct inference per mode

7. CONCLUSION AND FUTURE WORK

In this paper, a range of existing key detection algorithms was tested on datasets of audio and symbolic Irish music. Modifications of these models, and a new set of key-profiles, were introduced, which improved the performance on both types of representation. Error analysis showed that the most common confusions were between relative and neighbour keys, which reflects some specific characteristics of Irish music.

As stated in [9], it is sometimes difficult to pinpoint the key of an Irish tune. The key annotations on the datasets were made by the first author, and it is possible that other annotators could annotate some of the tunes differently. A way to quantify these ambiguities will be to gather annotations from other experienced musicians in order to obtain a Cohen's kappa coefficient. It is expected that most errors made by the key-matching algorithms will be on tunes where annotators disagree.

8. REFERENCES

[1] D. Temperley, *The cognition of basic musical structures.* Cambridge, Mass: MIT Press, 2001.

[2] E. Gómez, "Tonal Description of Polyphonic Audio for Music Content Processing," *INFORMS J. on Computing*, vol. 18, no. 3, pp. 294–304, Jan. 2006.

[3] S. T. Madsen, G. Widmer, and J. Kepler, "Key-Finding with Interval Profiles." in *ICMC*, 2007.

[4] K. Noland and M. B. Sandler, "Key Estimation Using a Hidden Markov Model." in *Proceedings of the 7th International Society for Music Information Retrieval Conference*, 2006, pp. 121–126.

[5] C. L. Krumhansl, *Cognitive Foundations of Musical Pitch*, ser. Oxford Psychology Series. Oxford, New York: Oxford University Press, 1990.

[6] F. Lerdahl, "Tonal Pitch Space," *Music Perception: An Interdisciplinary Journal*, vol. 5, no. 3, pp. 315–349, Apr. 1988.

[7] M. Leman, *Music and Schema Theory*, ser. Springer Series in Information Sciences, T. S. Huang, T. Kohonen, M. R. Schroeder, and H. K. V. Lotsch, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1995, vol. 31, dOI: 10.1007/978-3-642-85213-8.

[8] B. Duggan, "Machine annotation of traditional Irish dance music," Ph.D. dissertation, Dublin Institute of Technology, 2009.

[9] F. Vallely, Ed., *The Companion to Irish Traditional Music*, second edition ed. Cork: Cork University Press, 2011.

[10] F. Korzeniowski and G. Widmer, "Feature learning for chord recognition: the deep chroma extractor," in *Proceedings of the 17th International Society for Music Information Retrieval Conference*, New York, USA, 2016.

[11] J. D. Kelleher, B. Mac Namee, and A. D'Arcy, *Fundamentals of machine learning for predictive data analytics: algorithms, worked examples, and case studies.* Cambridge, Massachusetts: The MIT Press, 2015.

FERIN MARTINO'S TOUR: LESSONS FROM ADAPTING THE SAME ALGORITHMIC ART INSTALLATION TO DIFFERENT VENUES AND PLATFORMS

Jeffrey M. Morris
Texas A&M University
morris@tamu.edu

ABSTRACT

Ferin Martino is a piano-playing software algorithm created by Jeff Morris for art installations. It uses nearby motion observed with a camera to shape, but not dictate, its flow of musical decisions. Aesthetically, the work challenges the notions of the composer and the composition by presenting a software program that composes its own oeuvre in such a way that visitors cannot experience the composition without also influencing it. The installation has taken many forms, at times including multiple cameras and speakers, video display, note input by the visitor, a digital player piano, and an outdoor venue with an original sculpture embedded in nature. This algorithm has also proven useful in live performance and computer-aided composition. Case studies of exhibiting this work illustrate the process of finding the most effective venue for an interactive art installation and the process of tuning interactivity for a given venue.

1. INTRODUCTION

Ferin Martino is a piano-playing computer algorithm influenced by motion around it. motion of the viewers, seen by the computer's video camera, influences the intensity of the music created. Like David Cope's algorithm Emily Howell, [1] the striking character of the is algorithm inspired me to give it an anthropomorphic name even though this was not the original intent. Usually presented with the title, *The Collected Solo Piano Works of Ferin Martino, as Conjured by Your Presence*, this work creates a situation that lets us reflect on the ontological nature of music: this music cannot be heard without the audience causing changes in the composition: it is impossible to hear these "collected works" in "unadulterated" form; by listening, you are shaping the music. By extension, it offers a chance to reflect on the way that any composer's music only has its existence in the minds of its audiences, and that the modes of it existence may be as diverse as its listeners. This is an idea suggested by literary theorist Umberto Eco in *The Open Work* [2]. This work consists entirely of software and can generate new material practically indefinitely. The fact that the code fits on one screen indicates the elegance of the approach to generating endless music with pleasing results. This work is an example of something uncommon in technology-based art: it is acoustic computer music. The resulting music is acoustic or synthesized piano, in a capricious, expressive musical style that will not be overbearing for public spaces.

Ferin was not designed to be a "know-it-all" algorithm. Instead, it sets aside sophisticated computer vision to explore hat aesthetically-valuable subtleties can be captured and expressed by simply measuring the amount of motion seen between video frames. It initially presents itself as pure non-interactive spectacle. Given the chance, it will show the attentive visitor that motions can cause musical events to start or stop. The curious visitor is rewarded by discovering the seemingly playful nature of this interaction. The software does not simply convert actions into musical notes, but uses motion to disrupt its "train of thought." The result is a playful interaction partner that may tire of overactive visitors attempting to control it and may begin to please itself for a while. At other times, it may seem to amplify visitor's gestures or play rolling accompaniment instead of maintaining a predictable one-to-one behavior. Whereas it would seem to establish a direct link between motion and music, the experience turns out to be more complex. Since the video is essentially reduced to a single pixel for evaluation (the camera's view is not divided into "hot zones"), many gestures may achieve the same amount of overall motion, and the software is equally sensitive to unintended motions as well as clothing and background colors and changes in lighting (meaning it is somewhat weather-sensitive near windows). Because it uses a simple video camera, positioning the work in view of a window means that it can respond to motion inside and outside a given venue.

Initial experiences running the software in the background and monitoring it as I went about regular office work was quite rewarding. Because this work uses cameras to watch public areas, it can also be seen in relation to surveillance art. In this case, it presents itself as friendly and pleasant, but it can still be unnerving for a visitor to realize how their actions can be watched.

I've found the need to adjust the algorithm to best fit the traffic/motion patterns, distance from the traffic, and lighting, making its behavior in each venue unique. Its varied accompaniment to even mundane rituals can promote mindfulness throughout extended experiences in which visitors coexist in parallel with the software. As shown in the cases below, the addition of a video display, trackpad input, and digital player piano make it a more striking multi-sensory experience.

2. ABOUT THE SOFTWARE

Inspired by the free atonal expressionist music of Arnold Schoenberg, musical decisions begin with a choice between a single note treble melody and a three-note harmony below the melody. Pitch intervals are determined by a drunk walk, such that chord pitches are equally spaced according to that interval and are separated from the most recent melody pitch by that interval. A metronome emits a steady (but changeable) pulse triggering new note decisions. Video is converted into monochrome, the absolute differences of each pixel with the corresponding pixel in the previous frame are summed to one value representing overall motion, and that value scales the rate of the decision-triggering metronome as well as the loudness of the playing. This way, more motion yields more active music without giving the visitor direct one-to-one control over note decisions. [3]

Periodically, the software alternates between trigger by video and trigger by a running buffer of recent pitch choices. This loops and recursively shifts pitches to create florid musical sequences. The non-interactive aspect of this feature actually contributes to the appearance of organically developing musical motifs, because it is rare that a whole musical phrase would be captured in this buffer, more commonly just the tail of it that spins off in one direction or the other. It also contributes to the effect pf playfulness because it simply ignores the video at times, unannounced. Finally, it can be very rewarding to visitors who provide note input to it, as it may incorporate and elaborate their musical contributions.

For venues using a video display, the frame-differenced video is displayed with a blurring motion trail. This is helpful in getting visitors to interact with the software, because they can see what the software sees and where its attention flows. See Fig. 1.

3. CROWDED EXHIBIT HALL WITH HEADPHONES AND DISPLAY

The Triennale di Milano is a design and art museum in Milan, Italy, and since 1933, host of the triennial exhibition of the same name. [4] Ferin was positioned in an exhibition room where the 2013 Generative Art International Conference was held. Presented ‘shoulder to shoulder” with other works, it was necessary to use headphones. This, along with the compact arrangement of the venue, allowed for minimal movement, mostly waving. This tended to put visitors into a mindset of viewing paintings or a gallery of circus oddities: each is always already doing what it is supposed to do, and one feels as thought one might get “the point” of the work after only a few seconds. This presentation format made it uncomfortable for any visitors inclined to longer encounters to get to know the work better.

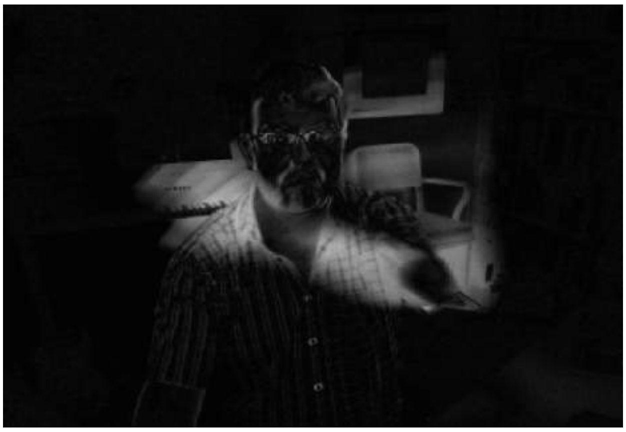


Figure 1. Camera input, processed and ready for analysis. This is what Ferin can see. The bright horizontal sweeping line represents the visitor moving a hand quickly from side to side.

4. MAIN LOBBY WITH SPEAKERS AND DISPLAY

The Onassis Cultural Center in Athens, Greece is a performing arts complex build in 2010 dedicated to promoting innovation in the arts. [5] Ferin was exhibited in the main lobby, facing the main entrance, as part of the 2014 joint Sound and Music Conference and International Computer Music Conference, with approximately 300 artist-scholars in attendance. It took the form of a laptop computer (using its built-in camera and display) and stereo speakers, all fit compactly into a 2 ft × 2 ft footprint encased in a custom wooden stand covered in black fabric.

Positioned against the back wall, the camera had a view of most of the lobby area, including an information desk used to register and assist conference participants; a seating area where participants met, talked, purchased beverages, and awaited scheduled performances; and heavy traffic on Syngrou Avenue outside the front door. To access the elevators, stairs, and concert venues, visitors walked directly up to the installation and continued around the left or right side of the building. Speakers were audible within the lobby, around the sides of the building, and upward one or two floors (before being drowned out by installations on other floors). It was not audible inside the concert venues.

This provided a steady range of motion seen by the camera (caused by cars or people), so the algorithm rarely normalized to the hyperactive level of video noise within the camera. This kept it from becoming too much of a distraction for a public meeting space. The experience was most effective for visitors on their way out of the building, because the connection between their movements and the music became quickly obvious as they suddenly entered the camera’s view then paused to look at the screen. This experience was more subtle for visitors entering the space, but it would respond to their gait as they made their way into the building. For these visitors, it made a subtle shift away from indifferent background music, gradually making noticeable connections with motions in the room.

In times when the lobby was empty, the software would normalize to the level of its video noise and become frenetic. This functioned nicely to gain the attention of people in nearby spaces; they knew something was happening over there without having to be there. Then, when they entered the camera's view and the camera normalized to their movement, the energetic burst followed by silence had the effect of a shy performer who didn't realize he had an audience.

This remains the most effective presentation scenario for the software alone, without user controls.

5. SIDE LOBBY WITH DIGITAL PLAYER PIANO

The Society for Electroacoustic Music in the United States (SEAMUS) 2015 conference was held in the Moss Arts Center, at Virginia Tech (Virginia Polytechnic Institute and State University). Ferin was placed in the atrium outside one of the concert venues. [6] This was a quiet, low-traffic area, so I felt a need to restrict its maximum activity so it wouldn't be too frenetic for the peaceful space. It proved to be a favorite lounge for some participants to relax. Since it was at a music conference, the addition of the Yamaha Disklavier digital player piano made engagement very rewarding for those that ventured to duet with the software. Besides the rewarding effect of hearing the software mirror and embellish one's own playing, the "struggle" over the keyboard is an especially exciting form of engagement. The visitor, not knowing when or what the software will play, is drawn into a hyper alert improvisation mindset involving intense critical listening as well as continually adjusted plans on when and what to play next. At times, visitors have noted that the software played notes just as the visitor was reaching for them (simultaneously frustrating and gratifying); other times, the software would operate in parallel accompaniment or musical dialogue with the visitor at the opposite end of the keyboard's range.

Even for viewers who didn't play the piano, the effect of watching a human interface (the keyboard) play automatically can evoke the image of a ghost playing the keyboard. They also got the thrill of watching the piano-playing visitors essentially become part of the artwork.

6. OUTDOORS WITH MULTIPLE CAMERAS AND SPEAKERS, WITHOUT DISPLAY

I-Park is a sculpture garden in East Haddam, Connecticut, focusing on ephemeral artworks engaging with nature. [7] For this exhibition, I cleared a circular path in an undeveloped part of the land. Fortuitous developments with other works on the property led me to install a sculpture built from parts of a grand piano and two upright pianos that had been left outdoors for years. I arranged them to resemble the carcass of a large mysterious beast and placed speakers in footlights surrounding it, to make it look like the overgrown site of a forgotten spectacle. Because of the circular path, I embedded four cameras in the sculpture, in order to capture motion in all directions from a suitable distance. I also use opportunity to develop a weather-proof housing for the computer and other electronics.

In order to more fully engage with the site, I incorporated another performing algorithm of mine that captured sounds from the environment, mixed them with the piano sound, and folded them together into rich contrapuntal textures. I originally intended to use plant and tree motion to influence Ferin, but there was a surprising dearth of wind at the time. As the demonstration video shows, this results in a very different character during the night compared to the day. It is busy at night because of increased wildlife activity as well as the software normalizing to the low light levels, in addition to the cameras' infrared illuminators changing the range of things that can be seen. During the day, it plays mildly as visitors walk into the area, exposed to the bright sun, but tended to "stare back" as visitors stopped to ponder the sculpture. It did tend to become more responsive to subtle motions of the visitors, like shifting their weight to one leg, repositioning arms, or turning heads.

For video documentation of this exhibition, see [8].

7. IMPROVISATION WITH SAXOPHONE

In a Disklavier-themed concert at Texas A&M University titled Hot-Wired Piano, I performed with Ferin and colleague Jayson Beaster Jones playing tenor saxophone. It was amusing to reflect on whether this was a duo or trio. Since I was there interacting with the software but not making any sound directly, this trio consisted of only two humans and two instruments, but one of the instruments wasn't being played by either of the humans. In a sense, the saxophone's sound was also (at times) influencing the software, because I replaced the keyboard input with pitch recognition.

At first, having simply replaced the keyboard input with pitch recognition, I quickly discovered that the pitch detection was so accurate that the algorithm would too-often play in unison with the saxophonist. While this was not a musical goal, and it wasn't conducive to building counterpoint, it also starkly highlighted expressive differences in intonation between the saxophone and the piano, which yielded undesirable sonic effects. In response, I added a gradually changing pitch offset between the pitch recognition and the note-triggering parts of the algorithm, and I created a key control that would momentarily unmute the audio input, so that I could have the algorithm listen to the saxophonist only at strategic moments.

With this unmute control in addition to the camera input, I functioned more like a conductor, musically flowing my hands in front of the camera to shape Ferin's musical flow. This type and range of motion felt most comfortable to my human physiology. It did loosely allow me to draw on my rudimentary conducting skills, even though the gestures bore no particular meaning to the software. Rather, my conducting experience informed my movements as a limited kind of dance: patterns that allowed me to sustain certain levels of continuous motion when desired and patterns to organize space to make room for quick, disjunct gestures.

For the audience, this experience charged the confusion and (hopefully) curiosity about which sounds and motions were causing which musical results, as well as tracing the life of some events as the are fed back in loops, changing between sounds and movements, influencing and sometimes echoing or elaborating on each other.

8. COMPUTER-ASSISTED COMPOSITION

Ferin has also been used successfully in computer-assisted composition [9]. In most cases, this workflow has looked much like a “duo” version of the live performance scenario, with me “conducting” in front of the camera and playing occasional brief motives on the keyboard to influence the software in desired directions. The latter has especially been useful in building large-scale form, since as of yet, the software has no long term memory. Left alone, motives only stay active in the music to the extent they happen to be captured in and played from the rolling buffer.

A computer-aided composition work session might typically involve a few “takes” like this and then turn to the editing stage. Given the software’s capricious rubato playing, automated transcription from MIDI to music notation is not straight-forward. It is helpful to create a MIDI reference track in a MIDI sequencer to accompany the recorded performance, by tapping out the basic metronome pulse where it should occur in the music, following the tempo as it changes. However, when a sequencer reinterprets the performance by mapping it to the MIDI reference track, all tempo information is lost from the resulting music notation, and needs to be manually interpreted into standard terms such as *allegro* and *ritardando*. Similar work us necessary for its rapid expressive dynamics as well.

This juncture, buried in tedium and practicality, is where Ferin Martino the composer is separated from Ferin Martino the performer. Musical nuances from Ferin’s performance are stripped away, leaving room for interpretation by the next (human) performer. As some of the *aura* (after Walter Benjamin [10]) of Ferin’s performance is striped away, the composition can gain new aura through its capacity for many variations in interpretations by other performers (cf. Eco [2]).

This algorithm’s output is surprisingly pianistic to the ear, especially for its relative simplicity (one page of code) and emergent nature, and curiously, it does not sound characteristic, or even very good, using any other instrument’s sound, even other keyboard instruments. One aspect contributing to its pianistic character must be its anthropomorphic design: with trichords played by a virtual left hand with five fingers, melodies played by a virtual right hand, pitch intervals and inter-onset (time) intervals kept within human scale, and “drunk walk” pseudorandom number generators to keep passages from being too disjunct. However, these design factors would make it just as good a fit for any keyboard instrument, including a harpsichord, organ, and electric piano, so there must be more to its good fit for the piano’s sound.

It seems that the algorithm’s output has been tuned to the modern pianoforte’s timbre (the spectrum and amplitude envelope) inadvertently to the exclusion of success with other timbres. Ferin Martino’s playing is often dense in pitch and in time. The harpsichord’s bright spectrum makes such passages sound too harsh, and temporal density only allows the attack transients to come through while masking the more attractive and interesting decay of each note. In contrast, the electric piano’s darker tone prevents individual notes from being discerned, often resulting in muddy sound masses. Organ tones result in similar results by their mostly flat amplitude envelopes, keeping note onsets from standing out, masked by the sustain of previous notes or with attention distracted by abrupt cutoffs—of course, organs are capable of quite lyrical legato playing, but Ferin’s emergent proclivity toward compound melodies (à la Hindemith) is made distractingly disjunct with organ-like amplitude envelopes. Experiments with mallet keyboard instrument sounds were more pleasing in regard to these factors, but they lose Ferin’s frequent expressive use of the damper pedal (and vibraphone suffers the same as the electric piano), besides the fact that it loses touch with human-scale playability on mallet instruments.

This highlights an important lesson in orchestration: with a long-established discrete pitch system, scales, a canon comprising music that heavily stresses melodies, chords, and human scale rhythms, and an abundance of theoretical analysis focusing on these parameters and not others, it is easy to imagine most instruments, especially the piano, as neutral, general purpose music generators, when really, each instrument’s idiosyncrasies shape the musical decisions of the composer, even if unbeknownst to the composer. Since Ferin Martino has no capacity for evaluating its own output, especially with regard to the spectral and temporal features described above, so this is a crucial stage in the development of a performer algorithm. This relates in spirit to William Sethares’s work in tuning systems, in which he demonstrates that consonance and dissonance are closely tied to timbre and tuning system more than Fortean interval class vectors. [11]

To finish the description of the transcription process: Ferin’s dense playing usually ends up yielding sheet music requiring advanced piano skill to play. This makes it very amenable to treating Ferin’s print out as a piano score to be orchestrated for small chamber ensembles, such as piano duo, piano trio (piano, violin, and cello), and string quartet. The manual orchestration process allows me (as the supervising composer, if not the one executing each note choice) to apply the considerations in the preceding paragraphs in order to map the musical content to fit each instrument as idiomatically effective as possible. The fact that Ferin’s playing sounds more naturally pianistic than it looks in sheet music is also telling; it suggests that the character of the piano-human combination has its own idiosyncrasies, a subset of the full characteristic possibilities of the piano itself (including other playing scenarios).

One such piece was premiered by the Apollo Chamber Players, a string quartet based in Houston, Texas. The main material was created in a single take with Ferin. I was sculpting Ferin’s behavior by moving my hands in front of the camera and occasionally introducing motives of a single repeated pitch via the piano keyboard. This created a dynamic in the piece of striving for stasis and falling away from it, as I would play repeated notes at a few certain junctures, and Ferin would copy parts of that motive and carry it off in sequences. After the initial performance, I distributed the voices for a string quartet. To build a more coherent form (without trying to cover up Ferin's capricious nature), I added a brief introduction comprising three slow cascading iterations of the repeated-pitch motive, and I inserted a partial recapitulation before the final sequence: at a point where the music seems stuck in a loop, I inserted a break, then a sequence of truncated clips from previous moments, in the order they happened, but always interrupted. This section is played *sul ponticello* to set it off as a copy of previous events, as if retracing one's steps before continuing. This recapitulation leads to the repeated motive where the original music was stopped, and it continues from that point in full voice, going on to the end. I gave it the title, *The Garden of Forking Paths* to reflect on the many musical sequences Ferin tests before moving on, as well as its ultimately through-composed winding path.

9. EXPERIMENTS WITH DANCE

Given these experiences with tuning the software to respond to human movement, it was natural to consider adapting this work for dance. However, these experiments quickly highlighted differences between a camera with video analysis and the human eye with human cognition. As dancers explored the full stage space, they would range from being so close to the camera that most of their bodies could not be seen to being so far at end of the stage that their whole bodies only occupied a small portion of the camera's view. A camera operator would naturally want to zoom in and out and pan around to keep the dancer's full body in the video frame. However, this revealed that moving the camera even slightly can yield a maximal amount of change in the video signal even when there is no movement on stage. Seeing the need for more development for this kind of performance, this exploratory performance was put on hold.

The problem of dancers' distance might be addressed by an automatic spatial normalizing process that crops out any space that contains no movement, leaving a tight-framed rectangle inside to evaluate. However, this will require many more decisions, thereby shaping the character of the work. Additionally, the quick pace of dancers' movement might be outside the software's scope of responsiveness that was effective for causal visitors. Making it more responsive may result in a less interesting one-to-one kind of interaction. The notion of camera operator as performer is an intriguing one worthy of future investigation, but it too will require many character-defining decisions. This line of inquiry may need to use a completely different performing algorithm from Ferin, but informed by these experiences.

10. EXPERIMENTS WITH PAINTING

Working with a painter for visual input has avoided the challenges described with dancers, since a painter is largely engaged with a two-dimensional vertical plane, like the camera and screen. April Zanne Johnson [12] already uses music and sound as a stimulus to influence her mark making vocabulary in her drawing and painting practice. She also sometimes thinks kinetically about the act of painting: making painting gestures in the air before and after touching the brush to the canvas in a process of contemplation, winding up, and following through. Both of these facts have made collaborations with Ferin Martino very fruitful. This creates a fascinating multi-modal creative feedback loop as she paints in response to the music and her motion while painting in turn shapes the music. Ferin's capacity for endless through-composed playing also suits this scenario well. Together, they produce an experience of painting and composition as a realtime process laid out across time, besides the final artifacts that result from the process.

11. OTHER FUTURE WORK

As highlighted throughout the discussion above, there are many areas where future developments can be done. Further work will investigate possibilities for building larger formal structures over time, including buffers on medium and long time scale, as well as memory cues as in Butch Morris's Conduction technique for conducting ensembles of improvisers. The software could also generate its own MIDI reference track for beat mapping during transcription. Further software routines could be created to automatically interpret tempo and dynamics changes into standard words and symbols. Synthetic timbres can be explored, sharing the key features of the piano and avoiding problematic features described above. Also in this vein, the software can be tuned to suit other instruments. This would be an especially interesting inquiry for thinking about composition, since it turns out that many of its most interesting melodic lines incorporate both virtual hands to build compound melodies or sequences. Finally, the software could be taken to another level of sophistication and self governance if it could monitor the sonic features of its own output and adjust its note decisions accordingly.

However, as these opportunities are opened up above, the discussion also highlights distinctive features of the character oft his algorithm that would be significantly changed, perhaps to become less intriguing or idiosyncratic, or at least becoming something that should be considered a different process altogether. These advances will be considered carefully with this potential sacrifice in mind. It will more likely spawn a number different performer agents rather than numerous upgrades to this one.

12. CONCLUSIONS

It is delightfully gratifying that something originally meant to be an afternoon project to demonstrate a Disklavier digital player piano, with a modicum of interactivity, for a tour of university dignitaries would provide so rich an output, become so fascinating a collaborator, open up deep questions of the nature of composition and performance, and reach audiences in Milan, Athens, and in the United States, Texas, the Southeast, the Mid-Atlantic, and New England.

These experiences show that simple, perhaps even simplistic solutions can reach new levels of elegance and open up new dimensions to contemplate, as with Alexander's solution to the Gordian knot and with Cantor's diagonal. In the end, an impatient attempt to simply "make music" has revealed myriad questions that are answered in that process, whether the composer takes conscious responsibility for them, leaves them to tacit intuition, or lets the governing technology fill in those answers according to what comes most naturally to it—for better or worse.

13. EXAMPLES

Video, audio, and musical scores resulting from this work can be found at [13].

Acknowledgments

Thanks to saxophonist Jayson Beaster Jones and painter April Zanne Jonhson for joining me in developing these productions, and to the I-Park Foundation and the Atlantic Center for the Arts for sponsoring this work.

14. REFERENCES

[1] D. Cope, *Computer Models of Musical Creativity*. Cambridge, MA, USA: MIT Press, 2005.

[2] U. Eco, *The Open Work*. Cambridge, MA, USA: Harvard University Press, 1962/1989.

[3] J. M. Morris, "Ferin Martino: A small piano algorithm and its lessons on creativity, interaction, and expression," in *3rd International Workshop on Musical Metacreation (MUME 2014)*. Artificial Intelligence and Interactive Digital Entertainment, 2014, pp. 40–44.

[4] T. di Milano, "History and mission," September 2016, http://www.triennale.org/en/chi-siamo/storia-e-mission/.

[5] A. S. Onassis Public Benefit Foundation, "Onassis cultural center athens," Decemeber 2015, http://www.onassis.org/en/cultural-center.php.

[6] J. Morris, "Ferin Martino demo," Video, September 2015, https://www.youtube.com/watch?v=iaDjezcrzHA.

[7] I. Park Foundation, "I-Park international artist-in-residence program," March 2003, http://www.i-park.org/.

[8] J. Morris, "Where Ferin was," November 2015, http://www.morrismusic.org/2015/where-ferin-was.

[9] J. M. Morris, "Collaborating with machines: Hybrid performances allow a different perspective on generative art," in *Proceedings of the Generative Art International Conference*. Polytechnic University of Milan, Italy, 2013, pp. 114–125.

[10] W. Benjamin, *Illuminations: Essays and Reflections*. Berlin, Germany: Schocken Books, 1936/1969, ch. The Work of Art in the Age of Mechanical Reproduction, pp. 217–252.

[11] W. A. Sethares, *Tuning, Timbre, Spectrum, Scale*, 2nd ed. New York, NY, USA: Springer, 2004.

[12] "Neurologically produced synesthetic forms and color: Studio visit with April Zanne Johnson," http://artmazemag.com/april-zanne-johnson/.

[13] J. Morris, "Ferin Martino," July 2014, http://www.morrismusic.org/ferinmartino.

EXPLORING MELODY AND MOTION FEATURES IN “SOUND-TRACINGS”

Tejaswinee Kelkar
University of Oslo, Department of Musicology
tejaswinee.kelkar@imv.uio.no

Alexander Refsum Jensenius
University of Oslo, Department of Musicology
a.r.jensenius@imv.uio.no

ABSTRACT

Pitch and spatial height are often associated when describing music. In this paper we present results from a sound-tracing study in which we investigate such sound–motion relationships. The subjects were asked to move as if they were creating the melodies they heard, and their motion was captured with an infra-red, marker-based camera system. The analysis is focused on calculating feature vectors typically used for melodic contour analysis. We use these features to compare melodic contour typologies with motion contour typologies. This is based on using proposed feature sets that were made for melodic contour similarity measurement. We apply these features to both the melodies and the motion contours to establish whether there is a correspondence between the two, and find the features that match the most. We find a relationship between vertical motion and pitch contour when evaluated through features rather than simply comparing contours.

1. INTRODUCTION

How can we characterize melodic contours? This question has been addressed through parametric, mathematical, grammatical, and symbolic methods. The applications of characterizing melodic contour can be for finding similarity in different melodic fragments, indexing musical pieces, and more recently, for finding motifs in large corpora of music. In this paper, we compare pitch contours with motion contours derived from people’s expressions of melodic pitch as movement. We conduct an experiment using motion capture to measure body movements through infra-red cameras, and analyse the vertical motion to compare it with pitch contours.

1.1 Melodic Similarity

Marsden disentangles some of our simplification of concepts while dealing with melodic contour similarity, explaining that the conception of similarity itself means different things at different times with regards to melodies [1]. Not only are these differences culturally contingent, but also dependent upon the way in which music is represented as data. Our conception of melodic similarity can be compared to the distances of melodic objects in a hyperspace of all possible melodies. Computational analyses of melodic similarity have also been essential for dealing with issues regarding copyright infringement [2], “query by humming” systems used for music retrieval [3, 4], and for use in psychological prediction [5].

1.2 Melodic Contour Typologies

Melodic contours serve as one of the features that can describe melodic similarity. Contour typologies, and building feature sets for melodic contour have been experimented with in many ways. Two important variations stand out — the way in which melodies are represented and features are extracted, and the way in which typologies are derived from this set of features, using mathematical methods to establish similarity. Historically, melodic contour has been analysed in two principal ways, using (a) symbolic notation, or (b) recorded audio. These two methods differ vastly in their interpretation of contour and features.

1.3 Extraction of melodic features

The extraction of melodic contours from symbolic features has been used to create indexes and dictionaries of melodic material [6]. This method simply uses signs such as +/-/=, to indicate the relative movement of each note. Adams proposes a method through which the key points of a melodic contour — the high, low, initial, and final points of a melody — are used to create a feature vector that he then uses to create typologies of melody [7]. It is impossible to know with how much success we can constrain melodic contours in finite typologies, although this has been attempted through these methods and others. Other methods, such as that of Morris, constrain themselves to tonal melodies [8], and yet others, such as Friedmann’s, rely on relative pitch intervals [9]. Aloupis et. al. use geometrical representations for melodic similarity search. Although many of these methods have found robust applications, melodic contour analysis from notation is harder to apply to diverse musical systems. This is particularly so for musics that are not based on western music notation. Ornaments, for example, are easier to represent as sound signals than symbolic notation.

Extraction of contour profiles from audio-based pitch extraction algorithms has been demonstrated in several recent studies [10, 11], including specific genres such as flamenco voice [12, 13]. While such audio-based contour extraction may give us a lot of insight about the musical data at hand,

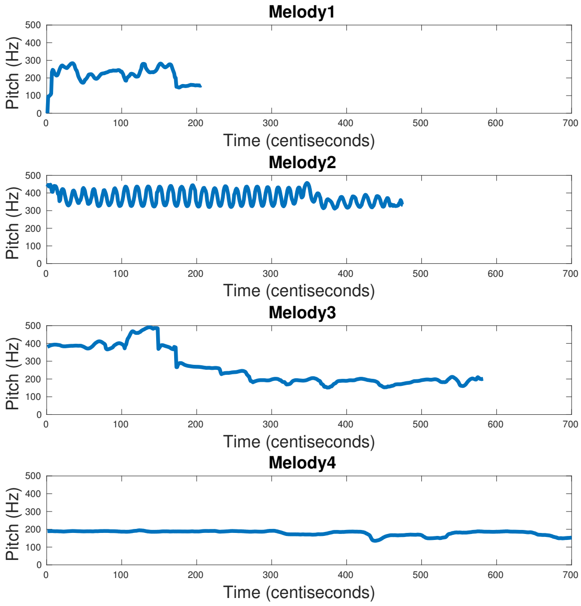


Figure 1. Examples of Pitch features of selected melodies, extracted through autocorrelation.

the generalisability of such a method is harder to evaluate than those of the symbolic methods.

1.4 Method for similarity finding

While some of these methods use matrix similarity computation [14], others use edit distance-based metrics [15], and string matching methods [16]. Extraction of sound signals to symbolic data that can then be processed in any of these ways is yet another method to analyse melodic contour. This paper focuses on evaluating melodic contour features through comparison with motion contours, as opposed to being compared to other melodic phrases. This would shed light on whether the perception of contour as a feature is even consistent, measurable, or whether we need other types of features to capture contour perception.

Yet another question is how to evaluate contours and their behaviours when dealing with data such as motion responses to musical material. Motion data could be transposed to fit the parameters required for score-based analysis, which could possibly yield interesting results. Contour extraction from melody, motion, and their derivatives could also demonstrate interesting similarities between musical motion and melodic motion. This is what this paper tries to address: looking at the benefits and disadvantages of using feature vectors to describe melodic features in a multimodal context. The following research questions were the most important for the scope of this paper:

1. Are the melodic contours described in previous studies relevant for our purpose?
2. Which features of melodic contours correspond to features extracted from vertical motion in melodic tracings?

In this paper we compare melodic movement, in terms of pitch, with vertical contours derived from motion capture recordings. The focus is on three features of melodic contour, using a small dataset containing motion responses of 3 people to 4 different melodies. This dataset is from a larger experiment containing 32 participants and 16 melodies.

2. BACKGROUND

2.1 Pitch Height and Melodic Contour

This paper is concerned with *melody*, that is, sequences of pitches, and how people trace melodies with their hands. Pitch appears to be a musical feature that people easily relate to when tracing sounds, even when the timbre of the sound changes independently of the pitch [17–19]. Melodic contour has been studied in terms of symbolic pitch [20, 21]. Eitan explores the multimodal associations with pitch height and verticality in his papers [22, 23]. Our subjective experience of melodic contours in cross cultural contexts is also investigated in Eerola's paper [24].

The ups and downs in melody have often been compared to other multimodal features that also seem to have up-down contours, such as words that signify verticality. This attribute of pitch to verticality has also been used as a feature in many visualization algorithms. In this paper, we focus particularly on the vertical movement in the tracings of participants, to investigate if there is, indeed, a relationship with the vertical contours of the melodies. We also want to see if this relationship can be extracted through features that have been explored to represent melodic contour. If the features proposed for melodic contours are not enough, we wish to investigate other methods that can be used to represent a common feature vector between melody and motion in the vertical axis. All 4 melodies in the small dataset that we create for the purposes of this experiment are represented as pitch in Figure 1.

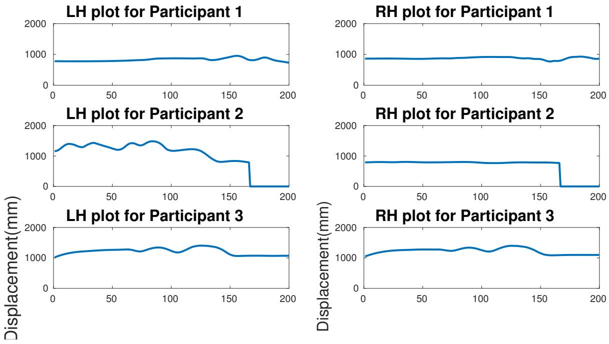


Figure 2. Example plots of some sound-tracing responses to Melody 1. Time (in frames) runs along the x-axes, while the y-axes represent the vertical position extracted from the motion capture recordings (in millimetres). LH=left hand, RH=right hand.

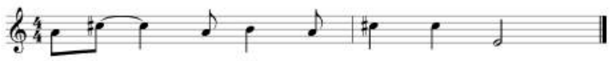


Figure 3. A symbolic transcription of Melody 1, a sustained vibrato of a high soprano. The notated version differs significantly from the pitch profile as seen in Figure 2. The appearance of the trill and vibrato are dimensions that people respond through in motion tracings, that don't clearly appear in the notated version.

	Feature 1	Feature 3
Melody1	[+, -, +, -, +, -],	[0, 4, -4, 2, -2, 4, 0, -9],
Melody2	[+, -, -]	[0, 2, -2, -2, 0, 0],
Melody3	[+, -, -, -, -, -, -],	[0, -2, -4, -1, -1, -1, -4, -2, -3, 0, 0, 0],
Melody4	[+, -, +, -, -, +, -, -]	[0, -2, 2, -4, 2, -2, 4, -2, -2]

Table 1. Examples of Features 1 and 3 for all 3 melodies from score.

2.2 Categories of contour feature descriptors

In the following paragraphs, we will describe how the feature sets selected for comparison in this study are computer. The feature sets that come from symbolic notation analysis are revised to compute the same features from the pitch extracted profiles of the melodic contours.

2.2.1 Feature 1: Sets of signed pitch movement direction

These features are described in [6], and involve a description of the points in the melody where the pitch ascends or descends. This method is applied by calculating the first derivatives of the pitch contours, and assigning a change of sign whenever the spike in the velocity is greater than or less than the standard deviation of the velocity. This helps us come up with the transitions that are more important to the melody, as opposed to movement that stems from vibratos, for example.

2.2.2 Feature 2: Initial, Final, High, Low features

Adams, and Morris [7, 8] propose models of melodic contour typologies and melodic contour description models that rely on encoding melodic features using these descriptors, creating a feature vector of those descriptors. For this study, we use the feature set containing initial, final, high and low points of the melodic and motion contours computed directly from normalized contours.

2.2.3 Feature 3: Relative interval encoding

In these sets of features, for example as proposed in Friedman, Quinn, Parsons, [6, 9, 14], the relative pitch distances are encoded either as a series of ups and downs, combined with features such as operators (¡,=,¿) or distances of relative pitches in terms of numbers. Each of these methods employs a different strategy to label the high and low points of melodies. Some rely on tonal pitch class distribution, such as Morris's method, which is also analogous to Schenkerian analysis in terms of ornament reduction; while others such as Friedmann's only encode changes that are relative to the ambit of the current melodic line. For the purposes of this study, we pick the latter method given as all the melodies in this context are not tonal in the way that would be relevant to Morris.

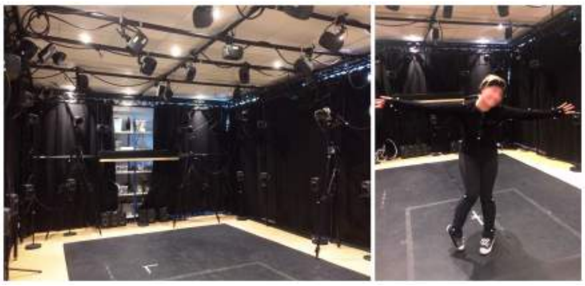


Figure 4. Lab set-up for the Experiment with 21 markers positioned on the body. 8 Motion capture cameras are hanging on the walls.

3. EXPERIMENT DESCRIPTION

The experiment was designed so that subjects were instructed to perform hand movements as if they were creating the melodic fragments that they heard. The idea was that they would "shape" the sound with their hands in physical space. As such, this type of free-hand sound-tracing task is quite different from some sound-tracing experiments using pen on paper or on a digital tablet. Participants in a free-hand tracing situation would be less fixated upon the precise locations of all of their previous movements, thus giving us an insight of the perceptually salient properties of the melodies that they choose to represent.

3.1 Stimuli

We selected 16 melodic fragments from four genres of music that use vocalisations without words:

1. Scat singing
2. Western classical vocalise
3. Sami joik
4. North Indian music

The melodic fragments were taken from real recordings, containing complete phrases. This retained the melodies in the form that they were sung and heard in, thus preserving their ecological quality. The choice of vocal melodies was both to eliminate the effect of words on the perception of music, but also to eliminate the possibility of imitating the sound-producing actions on instruments ("air-instrument" performance) [25].

There was a pause before and after each phrase. The phrases were an average of 4.5 seconds in duration (s.d. 1.5s). These samples were presented in two conditions: (1) the real recording, and (2) a re-synthesis through a sawtooth wave from an autocorrelation analysis of the pitch profile. There was thus a total of 32 stimuli per participant.

The sounds were played at comfortable listening level through a Genelec 8020 speaker, placed 3 metres ahead of the participants at a height of 1 meter.

3.2 Participants

A total of 32 participants (17 female, 15 male) were recruited to move to the melodic stimuli in our motion capture lab. The mean age of the participants was 31 years (*SD*=9). The participants were recruited from the University of Oslo, and included students, and employees, who were not necessarily from a musical background.

The study was reported to and obtained ethical approval from the Norwegian Centre for Research Data. The participants signed consent forms and were free to withdraw during the experiment, if they wished.

3.3 Lab set-up

The experiment was run in the fourMs motion capture lab, using a Qualisys motion capture system with eight wall-mounted Oqus 300 cameras (Figure 3.1, capturing at 200 Hz. The experiment was conducted in dim light, with no observers, to make sure that participants felt free to move as they liked. A total of 21 markers were placed on the body of the participants: the head, shoulders, elbows, wrists, knees, ankles, the torso, and the back of the body. The recordings were post-processed in Qualisys Track Manager (QTM), and analysed further in Matlab.

3.4 Procedure

The participants were asked to trace all 32 melody phrases (in random order) as if their hand motion was ‘producing’ the melody. The experiment lasted for a total duration of 10 minutes. After post processing the data from this experiment, we get a dataset for motion of 21 markers while the participants performed sound-tracing. We take a subset of this data for further analysis of contour features. In this step, we extract the motion data for the left and the right hands from a small subset of 4 melodies performed by 3 participants. We focus on the vertical movement of both the hands given as this analysis pertains to verticality of pitch movement. We process these motion contours along with the pitch contours for the 4 selected melodies, through 3 melodic features as described in section 2.2.

4. MELODIC CONTOUR FEATURES

For the analysis, we record the following feature vectors through some of the methods mentioned in section 1.2. The feature vectors are calculated as mentioned below:

Feature 1 Signed interval distances: The obtained motion and pitch contours are binned iteratively to calculate average values in each section. Mean vertical motion for all participants is calculated. This mean motion is then binned in the way that melodic contours are binned. The difference between the values of the successive bins is calculated. The sign of this difference is concatenated to form a feature vector composed of signed distances.

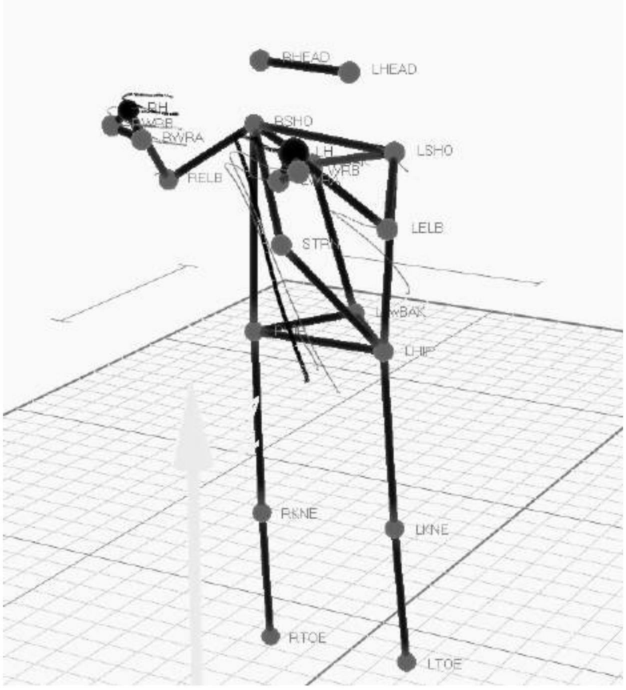


Figure 5. Example of post-processed Motion Capture Recording. The markers are labelled and their relative positions on the co-ordinate system is measured.

Feature 2 Initial, Final, Highest, Lowest vector: These features were obtained by calculating the four features mentioned above as indicators of the melodic contour. This method has been used to form a typology of melodic contours.

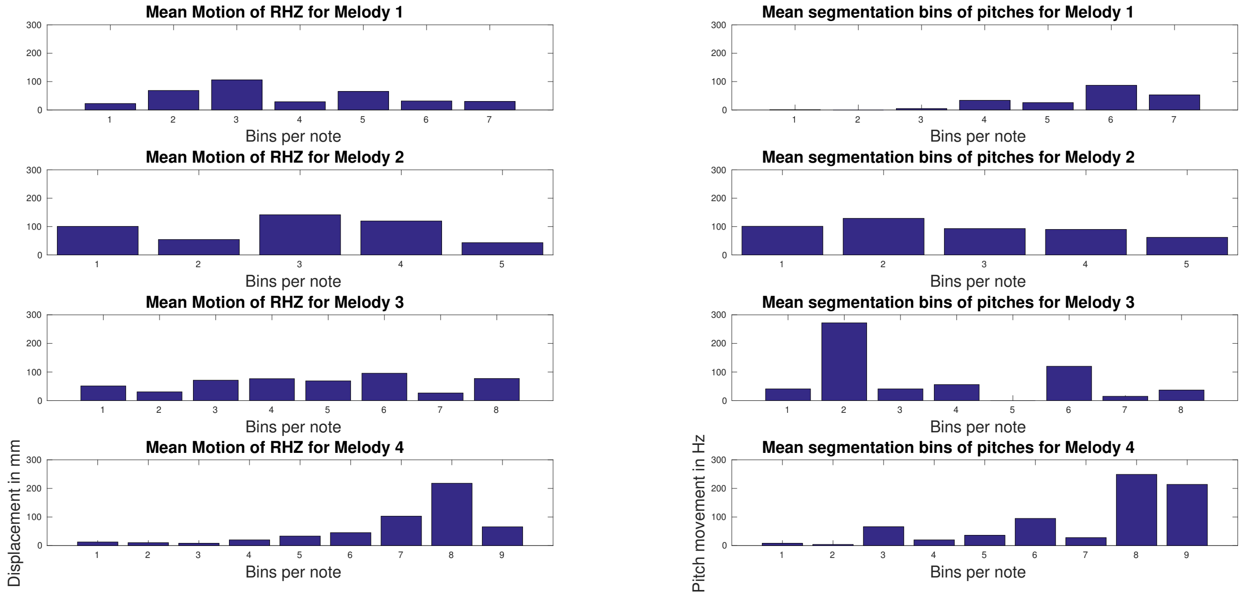
Feature 3 Signed relative distances: The obtained signs from Feature 1 are combined with relative distances of each successive bin from the next. The signs and the values are combined to give a more complete picture. Here we considered the pitch values at the bins. These did not represent pitch class sets, and therefore made the computation “genre-agnostic.”

Signed relative distances of melodies are then compared to signed relative distances of average vertical motion to obtain a feature vector.

5. RESULTS

5.1 Correlation between pitch and vertical motion

Feature 3, which considered an analysis of signed relative distances had the correlation coefficient of 0.292 for all 4 melodies, with a p value of 0.836 which does not show a confident trend. Feature 2, containing a feature vector for melodic contour typology, performs with a correlation coefficient of 0.346, indicating a weak positive relationship, with a p value of 0.07, which indicates a significant positive correlation. This feature performs well, but is not robust in terms of its representation of the contour itself, and fails when individual tracings are compared to melodies, yielding an overall coefficient of 0.293.



(a) Motion Responses

(b) Melodic Contour Bins

Figure 6. Plots of the representation of features 1 and 3. These features are compared to analyse similarity of the contours.

5.2 Confusion between tracing and target melody

As seen in the confusion matrix in Figure 7, the tracings are not clearly classified as target melodies by direct comparison of contour values itself. This indicates that although the feature vectors might show a strong trend in vertical motion mapping to pitch contours, this is not enough for significant classification. This demonstrates the need for having feature vectors that adequately describe what is going on in music and motion.

6. DISCUSSION

A significant problem when analysing melodies through symbolic data is that a lot of the representation of texture, as explained regarding Melody 2, gets lost. Vibratos, ornaments, and other elements that might be significant for the perception of musical motion can not be captured efficiently through these methods. However, these ornaments certainly seem salient for people's bodily responses. Further work needs to be carried out to explain the relationship of ornaments and motion, and this relationship might have little or nothing to do with vertical motion.

We also found that the performance of a tracing is fairly intuitive to the eye. The decisions for choosing particular methods of expressing the music through motion do not appear odd when seen from a human perspective, and yet characterizing what are significant features for this cross-modal comparison is a much harder question.

Our results show that vertical motion seems to correlate with pitch contours in a variety of ways, but most significantly when calculated in terms of signed relative values. Signed relative values, as in Feature 3, also maintain the context of the melodic phrase itself, and this is seen to be significant for sound-tracings. Interval distances matter less than the current ambit of melody that is being traced.

Other contours apart from pitch and melody are also significant for this discussion, especially timbral and dynamic changes. However, the relationships between those and motion were beyond the scope of this paper. The interpretation of motion other than just vertical motion is also not handled within this paper.

The features that were shown to be significant can be applied for the whole dataset to see relationships between vertical motion and melody. Contours of dynamic and timbral change can also be interesting to compare with the same methods against melodic tracings.

7. REFERENCES

[1] A. Marsden, "Interrogating melodic similarity: a definitive phenomenon or the product of interpretation?" *Journal of New Music Research*, vol. 41, no. 4, pp. 323–335, 2012.

[2] C. Cronin, "Concepts of melodic similarity in music copyright infringement suits," *Computing in musicology: a directory of research*, no. 11, pp. 187–209, 1998.

[3] A. Ghias, J. Logan, D. Chamberlin, and B. C. Smith, "Query by humming: musical information retrieval in an audio database," in *Proceedings of the third ACM international conference on Multimedia*. ACM, 1995, pp. 231–236.

[4] L. Lu, H. You, H. Zhang *et al.*, "A newapproach to query by humming in music retrieval." in *ICME*, 2001, pp. 22–25.

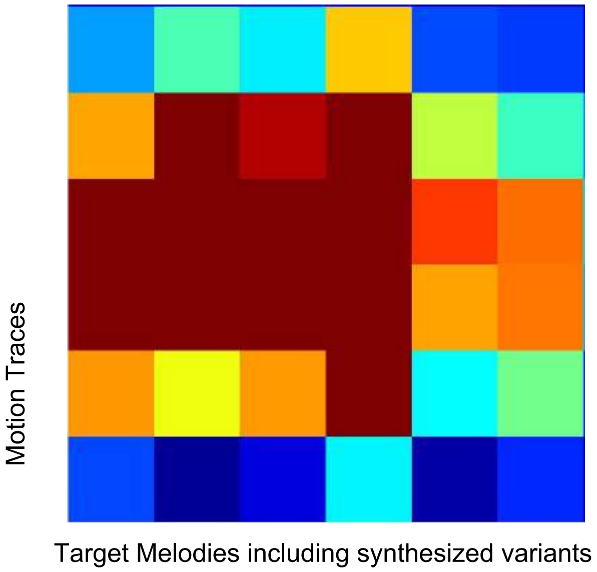


Figure 7. Confusion matrix for Feature 3, to analyse the classification of raw motion contours with pitch contours for 4 melodies.

[5] N. N. Vempala and F. A. Russo, "Predicting emotion from music audio features using neural networks," in *Proceedings of the 9th International Symposium on Computer Music Modeling and Retrieval (CMMR)*. Lecture Notes in Computer Science London, UK, 2012, pp. 336–343.

[6] D. Parsons, *The directory of tunes and musical themes*. Cambridge, Eng.: S. Brown, 1975.

[7] C. R. Adams, "Melodic contour typology," *Ethnomusicology*, pp. 179–215, 1976.

[8] R. D. Morris, "New directions in the theory and analysis of musical contour," *Music Theory Spectrum*, vol. 15, no. 2, pp. 205–228, 1993.

[9] M. L. Friedmann, "A methodology for the discussion of contour: Its application to schoenberg's music," *Journal of Music Theory*, vol. 29, no. 2, pp. 223–248, 1985.

[10] J. Salamon and E. Gómez, "Melody extraction from polyphonic music signals using pitch contour characteristics," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 6, pp. 1759–1770, 2012.

[11] R. M. Bittner, J. Salamon, S. Essid, and J. P. Bello, "Melody extraction by contour classification," in *Proc. ISMIR*, pp. 500–506.

[12] E. Gómez and J. Bonada, "Towards computer-assisted flamenco transcription: An experimental comparison of automatic transcription algorithms as applied to a cappella singing," *Computer Music Journal*, vol. 37, no. 2, pp. 73–90, 2013.

[13] J. C. Ross, T. Vinutha, and P. Rao, "Detecting melodic motifs from audio for hindustani classical music." in *ISMIR*, 2012, pp. 193–198.

[14] I. Quinn, "The combinatorial model of pitch contour," *Music Perception: An Interdisciplinary Journal*, vol. 16, no. 4, pp. 439–456, 1999.

[15] G. T. Toussaint, "A comparison of rhythmic similarity measures." in *ISMIR*, 2004.

[16] D. Bainbridge, C. G. Nevill-Manning, I. H. Witten, L. A. Smith, and R. J. McNab, "Towards a digital library of popular music," in *Proceedings of the fourth ACM conference on Digital libraries*. ACM, 1999, pp. 161–169.

[17] K. Nymoen, "Analyzing sound tracings: a multimodal approach to music information retrieval," in *Proceedings of the 1st international ACM workshop on Music information retrieval with user- centered and multimodal strategies*, 2011.

[18] M. B. Küssner and D. Leech-Wilkinson, "Investigating the influence of musical training on cross-modal correspondences and sensorimotor skills in a real-time drawing paradigm," *Psychology of Music*, vol. 42, no. 3, pp. 448–469, 2014.

[19] G. Athanasopoulos and N. Moran, "Cross-cultural representations of musical shape," *Empirical Musicology Review*, vol. 8, no. 3-4, pp. 185–199, 2013.

[20] M. A. Schmuckler, "Testing models of melodic contour similarity," *Music Perception: An Interdisciplinary Journal*, vol. 16, no. 3, pp. 295–326, 1999.

[21] J. B. Prince, M. A. Schmuckler, and W. F. Thompson, "Cross-modal melodic contour similarity," *Canadian Acoustics*, vol. 37, no. 1, pp. 35–49, 2009.

[22] Z. Eitan and R. Timmers, "Beethovens last piano sonata and those who follow crocodiles: Cross-domain mappings of auditory pitch in a musical context," *Cognition*, vol. 114, no. 3, pp. 405–422, 2010.

[23] Z. Eitan and R. Y. Granot, "How music moves," *Music Perception: An Interdisciplinary Journal*, vol. 23, no. 3, pp. 221–248, 2006.

[24] T. Eerola and M. Bregman, "Melodic and contextual similarity of folk song phrases," *Musicae Scientiae*, vol. 11, no. 1 suppl, pp. 211–233, 2007.

[25] R. I. Godøy, E. Haga, and A. R. Jensenius, "Playing air instruments: mimicry of sound-producing gestures by novices and experts," in *International Gesture Workshop*. Springer, 2005, pp. 256–267.

TROOP: A COLLABORATIVE TOOL FOR LIVE CODING

Ryan Kirkbride
University of Leeds
sc10rpk@leeds.ac.uk

ABSTRACT

Live Coding is a movement in electronic audiovisual performance that emerged at the turn of the millennia [1] and is now being performed all over the world through range of artistic practices [2]. It involves the continual process of constructing and reconstructing a computer program and, given that most performances are improvisational in nature [3], working together from multiple computers can provide a challenge when performing as an ensemble. When performing as a group Live Coders will often share network resources, such as a tempo clock to coordinate rhythmic information [4, 5], but rarely will they work together directly with the same material.

This paper presents the novel collaborative editing tool, Troop, that allows users to simultaneously work together on the same piece of code from multiple machines. Troop is not a Live Coding language but an environment that enables higher levels of communication within an existing language. Although written in Python to be used with the Live Coding language FoxDot [6] Troop provides an interface to other Live Coding languages, such as SuperCollider [7], and can be extended to include others. This paper outlines the motivations behind the development of Troop before discussing its applications as a performance tool and concludes with a discussion about the potential benefits of the software in a pedagogical setting.

1. INTRODUCTION

1.1 Collaborative Live Coding Environments

Rather than simply sharing the same clock when playing as a group Live Coders may choose to use a program that specifically facilitates a collaborative or synchronised performance. Such tools tend not to be programming languages themselves but software designed to aid communication between multiple computers using an existing Live Coding language. A useful example of this is the browser-based system Extramuros [8], which allocates each connected performer a small text box on a web page into which they can each write code. These text boxes are visible and can be edited by any other connected performer. Extramuros is "language-neutral" and can be used with any language that allows commands to be "piped" into it, which improves accessibility to a wider range of Live Coding practitioners.

Another browser-based tool for collaborative Live Coding is Gabber [9]; an extension to the Gibber library [10] that combines a chat-room interface and shared text buffers similar to Extramuros. Unlike Extramuros the edited code in each users' text buffers is only executed on that user's machine. Rohruber et al. [11] use an interface within SuperCollider similar to that of a chat-room to share small blocks of code dubbed "codelets". In contrast to Extramuros and Gabber, the "codelets" are shared with, but not executed on, each connected machine. Performers can choose to use or modify these chunks of code and submit them to the chat-room interface. This implementation stems from the performance style of PowerBooks Unplugged; the Live Coding ensemble discussed by Rohruber et al. [11], whose members sit at laptops at various points in the room and create a varying yet connected sonic experience for audiences. Since then Rohrhuber has gone on to develop a popular SuperCollider extension called The Republic [12] that allows performers connected over a network to access and modify one another's code. Similarly, LOLC is a "textual performance environment" that aims to facilitate methods of practice common to both improvisation and composition, with a focus on conversational communication [13]. It is a platform for sharing shorthand musical patterns, which are then played or transformed and re-shared by other participants.

As opposed to sending text between performers, Impromptu Spaces uses a tuple-space that acts as a "remote bulletin-board" for posting and retrieving information across a network [14]. This creates a shared and distributed memory that is accessible to each connected client and allows users to manipulate global variables such as tempo while avoiding any read/write clashes.

1.2 Motivation

Live Coding ensembles will often play connected over a network in order to synchronise their music and share information between one another. This presents several challenges for both the performers and audience during a live performance. One of the main technical challenges facing a Live Coding ensemble is that of temporal synchronisation. Latency (the time it takes for messages to be sent between computers) in network communication has to be accounted for when aligning multiple performers' computer clocks. When performing over large geographical distances a delay is affordable as long as the synchronisation between parts occurs at each end of the connection.

This style of synchronisation is used by Ogborn [15] and his network music duo, "Very Long Cat', that synchronises Live Coding and tabla drumming over the internet.

Alternatively, two or more Live Coders might perform together at the same location in front of an audience with each performers' laptop creating sound separately. Depending on the style of music this requires each machine to be temporally synchronised in real time, which is not always easy to accomplish. A common way to do this is through a clock-sharing protocol where each machine is constantly listening to a designated time-keeper, as implemented by the Birmingham Ensemble for Electroacoustic Research [4] and Algorave pioneers, Slub [5] for example. This requires an extra layer of configuration and is still liable to lag between performers' downbeat depending on the network being used.

Another resulting problem of this style of performance is the effect it has on the audience. Multiple Live Coders working on individual portions of code will usually have their work on separate screens, some, or all, of which will be projected for the audience. This can result in a non-optimal audience experience as there may either be too much going on or, if there aren't enough projectors available, not enough. One possible solution to this problem is the shared-buffer networked Live Coding system, Extramuros [8]. As mentioned previously this software uses a client-server model to connect multiple performers within a single web page. This allows them to create their own code and request and modify other performer's code in the same window, reducing the number of screens necessary to project during the performance. However, as the number of connected users, and consequently text boxes, increases the font size must be reduced and the code becomes less legible.

Gabber [9], the network performance extension to the JavaScript based language, Gibber, works in a similar way but also allows users to interact with each others code directly within the same text buffer. This allows for only one screen to be projected but all performers' work to be displayed. Originally Gabber used "a single, shared code editor" but it was found to be problematic "as program fragments frequently shifted in screen location during performances and disturbed the coding of performers". Gabber has since moved to a more distributed model. The single, shared text buffer model may have proved problematic in this instance but has seen much mainstream success, most notably in Google Docs [16], and has prompted me to create a similar tool for concurrent Live Coding collaboration entitled Troop.

2. DESIGN

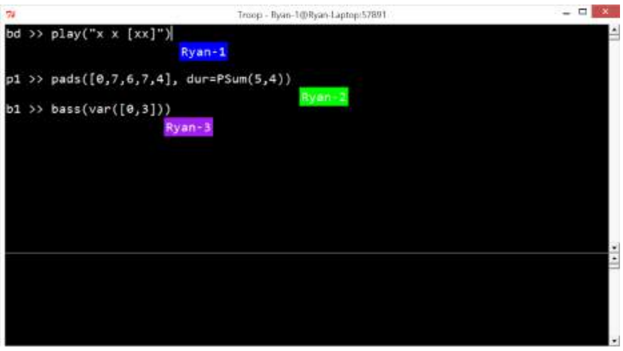
The purpose of the Troop software is to allow multiple Live Coders to collaborate within the same document in such a way that audience members will also be able to identify the changes made by the different performers. This section outlines the steps taken to achieve this and briefly describes the two types of network architecture used in this project.

2.1 Development Language

Troop is designed to work with the Live Coding language, FoxDot [6] but, like Extramuros, it can be language neutral with a small amount of configuration. The programming language Python [1] is useful for fast software development and comes with a built-in package for designing graphical user interfaces (GUIs) called `Tkinter`. As FoxDot is written in Python it makes it easy to pipe commands to its interpreter from Troop.

2.2 Interface

The philosophy behind Troop is that all performers seemingly share the same text buffer and contribute to its content at the same time. To allow each performer to differentiate their own contributions from others each connected performer is given a different coloured label which contains their name. This label's position within the input text box is mapped to the location of the respective performer's text cursor, as shown in Figure 1.



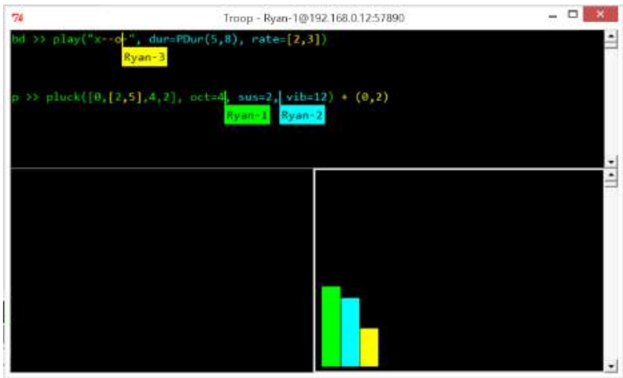
```
bd >> play("x x [xx]")
p1 >> pads([0,7,6,7,4], dur=PSum(5,4))
b1 >> bass(var([0,3]))
```

Figure 1. A screen shot of the first iteration of the Troop interface with three connected users.

As time elapses during a performance and the text buffer begins to fill up it is not always possible to separate the individual contributions and it becomes unclear as to whom has written what; even to the performers. This problem has been identified by Xambó et al. [17], stating there is a challenge in "identifying how to know what is each others code, as well as how to know who has modified it".

To help combat this problem and differentiate each performers' contributions each connected performer is allocated a different colour for text (see Figure 2) and highlighting. By doing this performers can leave traces of their own coloured code throughout the communal text buffer. Editing a block of another performer's code interweaves their colours and thought processes, creating a lasting visual testament to a collaborative process; or at least until someone else makes their own edit. The colour of the text entered by each performer matches the colour of their marker to retain a consistency in each performers' identity. It is customary in Live Coding for evaluated code to be temporarily highlighted and by doing this in separate colours it allows both audience and performers to identify the source of that action.

[1] `http://python.org/`



```
bd >> play("x--o-", dur=PDur(5,8), rate=[2,3])

p >> pluck([0,[2,5],4,2], oct=4, sus=2, vib=12) + (0,2)
```

Figure 2. A screen shot of the second iteration of the Troop interface with three connected users.

Performers contributions are also measured in terms of quantity of characters present. In the bottom right corner of 2 there is a bar-chart that displays the proportion of code contributed by each performer. As a collaborative performance tool it is particularly interesting to keep track of this information and use it as a creative constraint for a performance (see section 4.2.1 for a discussion on this).

2.3 Network Architecture

The Troop software uses a client-server model but can be set up in two ways based on which machine will be handling code execution.

2.3.1 Server-Side Execution Model

This first design uses a centralised synchronous networked performance model similar to that of LOLC [13] and transfers messages using TCP/IP to ensure all messages are sent over the network. The server runs on a machine dedicated to listening to client applications (see figure 3), which are sending keystroke information, and handles the code execution and sound creation. Incoming keystroke data are stored in a queue structure to avoid sending data sent from two clients simultaneously in differing orders to other clients. As the number of connected clients increases so too would the chance of this error occurring. A disadvantage of storing all keystroke data in a queue on the server is that there is sometimes a noticeable latency between pressing a key and the corresponding letter appearing the text editor depending on the network.

In this topology only the server machine executes code and schedules musical events, which eliminates any need for clock synchronisation. This method also utilises a centralised shared name-space, which doesn't require information about the code to be sent to the client applications. The disadvantage to this approach is that all performers must be "local" to the server (a client and server can be run on the same machine together) if they want to hear the performance, unless they stream the audio through another program from the server machine.

2.3.2 Client-Side Execution Model

When performers are not co-located it might be required that code is run on the client machines as opposed to the server. In this case, the *Troop* server is run with an extra command-line argument:

```
./run-server.py --client
```

This signals to the server to send information about what code to evaluate back to the client. Figure 4 shows how the code execution and sound creation are shifted to the client's responsibility in this set up. A client-server model was adopted over a peer-to-peer topology as it meant that the client application did not have to be reconfigured based on where the code was being executed and the centralised server queue structure for storing keystrokes could still be used.

The nature of Troop's concurrent text editing means that code in each client's text buffer is identical. This means that there is no need to use a shared name-space, such as in [14, 18], as the data is reproduced on each connected machine. An advantageous consequence of this is that Troop need only send raw text across the network and no serialising of other data types is required in order for the program to be reproduced on multiple machines.

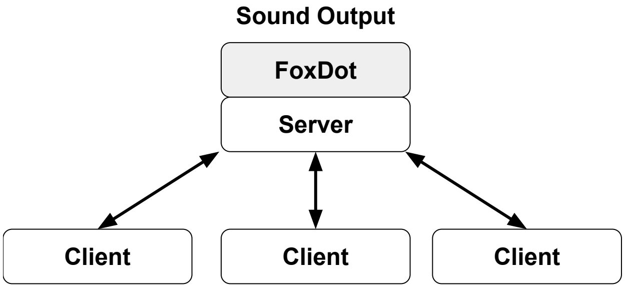


Figure 3. Network diagram for Troop's server-side execution model.

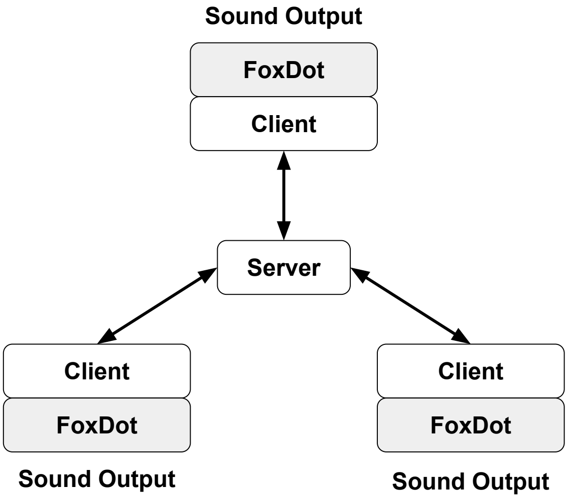


Figure 4. Network graph of *Troop*'s client-side execution model.

2.4 Extending the software for other languages

Troop contains a module called `interpreter.py` that defines the classes used for communicating with host Live Coding languages. Currently there are interpreter classes for the FoxDot and SuperCollider languages. If a language can take a raw string input and convert that to usable code then Troop will be able to communicate with it. Specifying the `lang` flag at the command line, followed by the name of the desired language, will start the client with a different interpreter to FoxDot like so:

```
./run-client.py --lang SuperCollider
```

By giving performers flexible access to multiple Live Coding languages Troop can enhance musical expressivity and broaden the channels for collaboration.

3. SYNCHRONISATION

3.1 Maintaining Text Consistency

One of the most difficult aspects to this project was making sure that the contents of the text buffer for each connected client remained identical. In Troop's first iteration, text would be added immediately on the local client then sent to the server to be forwarded to the other connected clients. Problems arose mainly when two clients were editing the same line, which caused the characters to be in different orders for each client. To combat this the clients in the next version of Troop would send all keystroke information to the server, which dispersed them to each client in the same order. This worked very well if the server application was running on the local machine but there were noticeable delays between key-press and text appearing on screen when connecting to a public server running Troop in a London data centre.

A solution was to combine these two input systems to maximise efficiency and still maintain consistency between client's text. When a key is pressed, Troop checks if there are any other client's currently editing the same line as the local client and if there are, it sends all the information to server to make sure the characters are entered in the same order for every client. If the local client is the only one editing a line the character is added immediately on the local machine and then sent to the server to tell other clients to add the character. This reduces latency for character inputs when clients are not interfering with one another but also maintains consistency at times when they are.

3.2 Limiting Random Values

When random number generators are utilised to define pitches and rhythms their values may become inconsistent between each machine and each performer will consequently hear a different piece of music. This can be combated by selecting the random number generator's seed at the start of the performance. In FoxDot for example, evaluating the command `Random.seed(1)` will do so for all connected clients and will consequently mean that any random elements will, although appear random, be completely deterministic for that session.

3.3 Clock Synchronisation

Along with information about each client's keystrokes, the Troop server sends a "ping" message to each connected client once every second. This serves two purposes; the first is to periodically check that each client can still be contacted and remove those that cannot from its address book, and the second is to aid clock synchronisation. A consequence of this "ping" message is that the `settime` method of Troop's interpreter class (Section 2.4) is called with a value, which can then communicate with whatever time-keeping data structure is used by the host language if need be.

4. CONCLUSIONS

This paper has presented the Live Coding environment, Troop, and discussed its potential as a tool for enhancing channels for collaboration and communication in Live Coding ensembles.

4.1 Applications in Teaching

Music has been used as a useful teaching analogy for computer science in younger age groups, as demonstrated by software such as Scratch [19] and the Live Coding language Sonic-Pi [20], and collaborative Live Coding can help move student thinking "from an individual to a social plane" [17].

By running several Troop servers in a classroom environment Troop would encourage collaborative work between students and allow a single teacher to monitor the work of multiple groups of students from a single machine.

4.2 Future Extensions

4.2.1 Adding Creative Constraints

Troop is an environment that facilitates group work over a network but each contributor in a document still maintains their own identity within the process through the use of coloured code. Maintaining this identity requires the number of characters entered by each user to be stored. This information could be used to add a constraint to a particular session to explore different collaborative approaches to Live Coding.

On each keystroke it would be possible to test a constraint and then only add the character to the text buffer if that test returns true. An example of this might be to disallow a single user to take up more than 50% of the text buffer, which could easily be achieved using the short function outlined in pseudo-code in Figure 5. The start of the performance would be difficult as performers would have to work together slowly to make sure they don't spend long periods of time waiting for other users to catch up, but as the total number of characters in the text buffer increased so too would the flexibility for writing larger portions of code in one go.

Another example might be to "ban" a random user from typing for a number of seconds to try and force them to look at what the rest of their ensemble are doing, or to only allow them to edit a single line of code written by one

```
def constraint(text, client):
    if client.chars > text.chars / 2:
        return False
    else:
        return True
```

Figure 5. Example pseudo-code for a creative constraint for Troop.

of their co-performers. The next iteration of Troop will implement a `constraints.py` module that will contain several creative constraints and allow users to define their own functions.

4.2.2 Web Browser Version

Live Coding environments such as Gibber [10] and LiveCodeLab [21] run in most standard web browsers and require no installation on the client machine. This lowers the barriers to entry and also makes the application platform-independent (provided the web browser itself runs correctly). Moving Troop to a Javascript based client could help provide a simpler set up procedure for those newer to computer programming.

4.3 Evaluation

Currently a small group of performers are learning to use the FoxDot and Troop software in order to perform at an event at the end of April 2017. Feedback from both the users and audience members will be collected, assessed, and contribute to the further development of Troop.

Acknowledgments

This research is funded by the White Rose College of Arts and Humanities (`http://www.wrocah.ac.uk`).

5. REFERENCES

[1] N. Collins, A. McLean, J. Rohrhuber, and A. Ward, "Live coding in laptop performance," *Organised sound*, vol. 8, no. 03, pp. 321–330, 2003.

[2] TOPLAP, "TOPLAP — the home of live coding," http://toplap.org/, 2004, accessed: 08/12/16.

[3] T. Magnusson, "Herding cats: Observing live coding in the wild," *Computer Music Journal*, vol. 38, no. 1, pp. 8–16, 2014.

[4] S. Wilson, N. Lorway, R. Coull, K. Vasilakos, and T. Moyers, "Free as in beer: Some explorations into structured improvisation using networked live-coding systems," *Computer Music Journal*, vol. 38, no. 1, pp. 54–64, 2014.

[5] A. McLean, "Reflections on live coding collaboration," *Proceedings of the Third Conference on Computation, Communication, Aesthetics and X. Universidade do Porto, Porto*, p. 213, 2015.

[6] R. Kirkbride, "FoxDot: Live coding with python and supercollider," in *Proceedings of the International Conference of Live Interfaces*, 2016, pp. 194–198.

[7] J. McCartney, "Rethinking the computer music language: Supercollider," *Computer Music Journal*, vol. 26, no. 4, pp. 61–68, 2002.

[8] D. Ogborn, "d0kt0r0/extramuros: language-neutral shared-buffer networked live coding system," https://github.com/d0kt0r0/extramuros, 2016, accessed 13/12/16.

[9] C. Roberts, K. Yerkes, D. Bazo, M. Wright, and J. Kuchera-Morin, "Sharing time and code in a browser-based live coding environment," in *Proceedings of the First International Conference on Live Coding*. Leeds, UK: ICSRiM, University of Leeds, 2015, pp. 179–185.

[10] C. Roberts and J. Kuchera-Morin, *Gibber: Live coding audio in the browser*. Ann Arbor, MI: Michigan Publishing, University of Michigan Library, 2012.

[11] J. Rohrhuber, A. de Campo, R. Wieser, J.-K. van Kampen, E. Ho, and H. Hölzl, "Purloined letters and distributed persons," in *Music in the Global Village Conference (Budapest)*, 2007.

[12] A. de Campo and J. Rohrhuber, "Republic," https://github.com/supercollider-quarks/Republic, accessed: 05/02/2017.

[13] J. Freeman and A. Van Troyer, "Collaborative textual improvisation in a laptop ensemble," *Computer Music Journal*, vol. 35, no. 2, pp. 8–21, 2011.

[14] A. C. Sorensen, "A distributed memory for networked livecoding performance," in *Proceedings of the ICMC2010 International Computer Music Conference*, 2010, pp. 530–533.

[15] D. Ogborn, "Live coding together: Three potentials of collective live coding," *Journal of Music, Technology & Education*, vol. 9, no. 1, pp. 17–31, 2016.

[16] Google, "Google docs - create and edit documents online, for free," https://www.google.co.uk/docs/about/, 2017, accessed: 02/02/17.

[17] A. Xambó, J. Freeman, B. Magerko, and P. Shah, "Challenges and new directions for collaborative live coding in the classroom," 2016.

[18] S. W. Lee and G. Essl, "Models and opportunities for networked live coding," in *Live Coding and Collaboration Symposium*, vol. 1001, 2014, pp. 48 109–2121.

[19] A. Ruthmann, J. M. Heines, G. R. Greher, P. Laidler, and C. Saulters II, "Teaching computational thinking through musical live coding in scratch," in *Proceedings of the 41st ACM technical symposium on Computer science education*. ACM, 2010, pp. 351–355.

[20] S. Aaron, “Sonic pi performance in education, technology and art,” *International Journal of Performance Arts and Digital Media*, vol. 12, no. 2, pp. 171–178, 2016.

[21] D. Della Casa and G. John, “Livecodelab 2.0 and its language livecodelang,” in *Proceedings of the 2nd ACM SIGPLAN international workshop on Functional art, music, modeling & design*. ACM, 2014, pp. 1–8.

AN ADAPTIVE KARAOKE SYSTEM THAT PLAYS ACCOMPANIMENT PARTS OF MUSIC AUDIO SIGNALS SYNCHRONOUSLY WITH USERS' SINGING VOICES

Yusuke Wada **Yoshiaki Bando** **Eita Nakamura** **Katsutoshi Itoyama** **Kazuyoshi Yoshii**
Department of Intelligence Science and Technology
Graduate School of Informatics, Kyoto University, Japan
{wada, yoshiaki, enakamura, itoyama, yoshii}@sap.ist.i.kyoto-u.ac.jp

ABSTRACT

This paper presents an adaptive karaoke system that can extract accompaniment sounds from music audio signals in an online manner and play those sounds synchronously with users' singing voices. This system enables a user to expressively sing an arbitrary song by dynamically changing the tempo of the user's singing voices. A key advantage of this systems is that users can immediately enjoy karaoke without preparing musical scores (MIDI files). To achieve this, we use online methods of singing voice separation and audio-to-audio alignment that can be executed in parallel. More specifically, music audio signals are separated into singing voices and accompaniment sounds from the beginning using an online extension of robust nonnegative matrix factorization. The separated singing voices are then aligned with a user's singing voices using online dynamic time warping. The separated accompaniment sounds are played back according to the estimated warping path. The quantitative and subjective experimental results showed that although there is room for improving the computational efficiency and alignment accuracy, the system has a great potential for offering a new singing experience

1. INTRODUCTION

Karaoke is one of the most popular ways of enjoying music in which people can sing their favorite songs synchronously with musical accompaniment sounds prepared in advance. In the current karaoke industry, musical scores (MIDI files) are assumed to be available for generating accompaniment sounds. Professional music transcribers are therefore asked to manually transcribe music every time new commercial CD recordings are released. The critical issues of this approach are that music transcription is very time-consuming and technically demanding and that the quality of accompaniment sounds generated from MIDI files is inferior to that of original musical audio signals.

It is impractical for the conventional approach to manually transcribe a huge number of songs on the Web. Consumer generated media (CGM) has recently been become more and more popular and many non-professional people have composed and distributed their own original songs on the Web. In Japan, for example, over 120 thousand songs have been uploaded on a media sharing Web service from July 2007 [1]. It is thus necessary to generate high-quality accompaniment sounds from *arbitrary* music audio signals without using musical scores or lyrics.

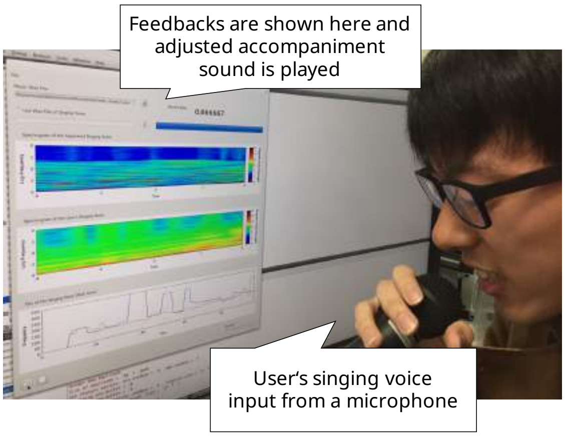

Feedbacks are shown here and adjusted accompaniment sound is played
User's singing voice input from a microphone

Figure 1. An example of how to use the proposed system. A user is allowed to expressively sing a song while accompaniment sounds are played back synchronously with the user's singing voices. The spectrograms and F0 trajectories of the user's singing voices and those of the time-stretched original singing voices can be compared in real time. The progress of singing voice separation is also displayed.

Another limitation of the current karaoke systems is that users need to manually set the tempo of accompaniment sounds in advance. Although this limitation can be acceptable for standard popular music with steady tempo, some kinds of music (e.g., opera, gospel, and folk songs) are usually sung in an expressive way by dynamically changing the tempo of the music.

To solve these problems, we propose an adaptive karaoke system that can extract accompaniment sounds from music audio signals in an online manner and play those sounds synchronously with users' singing voices.[1] Figure 1 shows how to use the proposed system. Once a song is selected, a user is allowed to immediately start to sing the song while listening to adaptively played-back accompaniment sounds separated from music audio signals. If the user gradually

[1] A demo video of the proposed system is available online: http://sap.ist.i.kyoto-u.ac.jp/members/wada/smc2017/index.html

accelerates (decelerates) the singing, the tempo of accompaniment sounds is accelerated (decelerated) accordingly such that those sounds are synchronized with the user's singing. The pitches (fundamental frequencies, F0s) of the singing voices can be compared with those of original singing voices in real time. To use this system, all the user has to do is to prepare only music audio signals.

This system mainly consists of three components: karaoke generation based on singing voice separation, audio-to-audio singing-voice alignment, and time-stretching of accompaniment sounds (Figure 2). More specifically, accompaniment sounds are separated from music audio signals using an online extension of robust nonnegative matrix factorization (RNMF) [2]. The stretch rate of the separated accompaniment sounds is estimated using online dynamic time warping (DTW) between a user's and original singing voices. Finally, the stretched version of the accompaniment sounds is played back. Since these steps are in parallel, it can therefore conceal the processing time for the singing voice separation from the user.

The main technical contribution of this study is to tackle real-time audio-to-audio alignment between singing voices whose pitches, timbres, and tempos many significantly vary over time. Note that conventional studies on singing-voice alignment focus on alignment between singing voices and symbolic information such as musical scores or lyrics. Another contribution is to apply this fundamental technique to a practical application of music performance assistance.

2. RELATED WORK

This section reviews related work on singing information processing and automatic accompaniment.

2.1 Karaoke Systems

Tachibana *et al.* [3] proposed a karaoke system that generates accompaniment sounds from input music audio signals without preparing musical scores or lyrics. This system uses a voice suppression technique to generate the accompaniment sounds, whose pitches can be changed manually. Inoue *et al.* [4] proposed another karaoke system that automatically adjusts the tempo of accompaniment sounds to a user's singing voices, assuming that musical scores and lyrics are prepared in advance.

2.2 Automatic Music Accompaniment

There have been many studies on automatic music accompaniment [5–11]. Dannenberg [5] proposed an online algorithm based on dynamic programming for automatic accompaniment. Vercoe [6] proposed an accompaniment system that supports live performances using traditional musical instruments. Raphael [7] used a hidden Markov model (HMM) to find optimal segmentation of the musical score of a target musical piece. Cont [8] designed an architecture that features two coupled audio and tempo agents based on a hybrid hidden Markov/semi-Markov framework. Nakamura *et al.* [9] reduced the computational complexity of polyphonic MIDI score following using an outer-product HMM. Nakamura *et al.* [10] also proposed an efficient score-following algorithm under an assumption that the prior distributions of score positions before and after repeats or skips are independent of each other. Montecchio and Cont [11] proposed a particle-filter-based method of real-time audio-to-audio alignment between polyphonic audio signals without using musical scores.

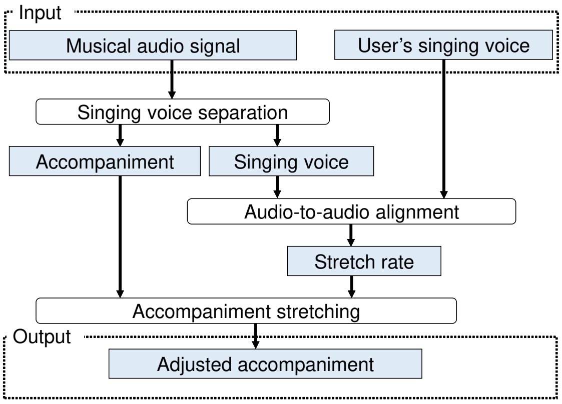


Figure 2. An overview of the system implementation.

2.3 Singing Voice Alignment

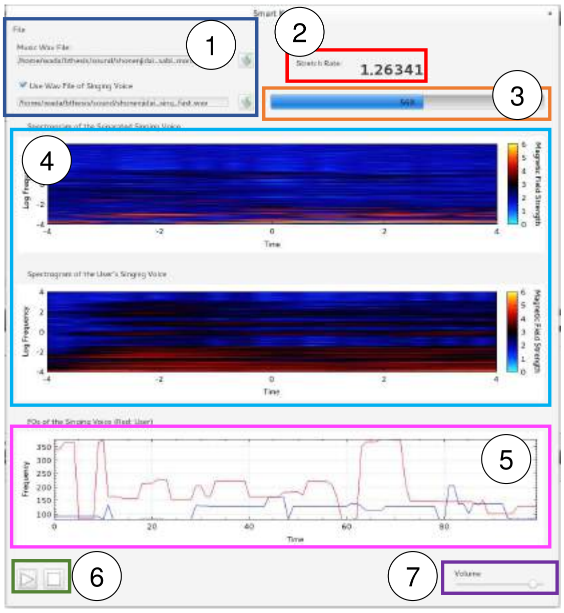
Many studies have addressed audio-to-score or audio-to-lyric alignment, where singing voices are aligned with symbolic data such as musical scores or lyrics [12–15]. Gong *et al.* [12] attempted audio-to-score alignment based on a hidden semi-Markov model (HSMM) using melody and lyric information. A lot of effort has been devoted to audio-to-lyric alignment. Fujihara *et al.* [13], for example, used singing voice separation and phoneme alignment for synchronizing musical audio signals with their corresponding lyrics. Iskandar *et al.* [14] attempted syllabic-level alignment based on dynamic programming. Wang *et al.* [15] combined feature extraction from singing voices rhythmic structure analysis of musical audio signals. Dzhambazov *et al.* [16] modeled a duration of each phoneme based on a duration-explicit HMM using mel-frequency cepstral coefficients (MFCCs).

2.4 Singing Voice Separation

A typical approach to singing voice separation is to estimate a time-frequency mask that separates the spectrogram of a target music audio signal into a vocal spectrogram and an accompaniment spectrogram [17–20]. Huang *et al.* [17] used robust principal component analysis (RPCA) to extract accompaniment spectrograms with low-rank structures. Deep recurrent neural networks were also used [21]. Ikemiya *et al.* [18] improved the separation quality by combining RPCA with F0 estimation. Rafii and Pardo [19] proposed a similarity-based method to find repetitive patterns (accompaniment sounds) in polyphonic audio signals. As another approach, Yang *et al.* [20] used Bayesian non-negative matrix factorization (NMF). Very few studies have been conducted on online singing voice separation.

3. PROPOSED SYSTEM

This section describes the graphical user interface (GUI) of the proposed system and the implementation of the system based on singing voice separation and audio-to-audio alignment between singing voices.


Smart K
File
Music Wav File
☑ Use Wav File of Singing Voice
Spectrogram of the Separated Singing Voice
Spectrogram of the User's Singing Voice
F0s of the Singing Voice (Red: User)
Log Frequency
Time
Magnetic Field Strength
Frequency
Volume
1
2
3
4
5
6
7

Stretch Rate 1.26341

Figure 3. A screenshot of the user interface.

3.1 User Interface

Figure 3 shows a screenshot of the user interface that provides easy-to-use functions through seven GUI elements: (1) a selector for music audio signals, (2) a display of the current stretch rate, (3) a display of the progress of singing voice separation, (4) a display of the spectrograms of the user's and the original singing voices, (5) a display of the F0 trajectories of the user's and the original singing voices, (6) play and stop buttons for controlling the playback, and (7) a volume control for the accompaniment sound.

The GUI elements numbered 2, 4, and 5 provides visual feedback of the user's singing voice and the original singing voice. The red area (number 2 in Figure 3) indicates whether the stretch rate is matching the user's intention. The user can refer how the original singer sings with the spectrograms displayed in the sky blue area (number 4 in Figure 3). For example, the user can know sections in which the original singer users a vibrato technique. In addition, the F0 trajectories displayed in the pink area (number 5 in Figure 3) helps the user to correct the pitch of the user's singing voice.

3.2 Implementation Policies

To reduce the user's wait time, we specify three requirements for the system implementation. First, users should be able to enjoy karaoke immediately after starting the system. Second, singing voice separation should to be processed in real time without prior learning. Third, automatic accompaniment should also be processed in real time.

We chose and implemented a method for each component of the system so as to satisfy these three requirements. More specifically, singing voice separation, recording of a user's singing voices, singing-voice alignment, and playback of time-stretched accompaniment sounds are processed in independent threads (Figure 2).

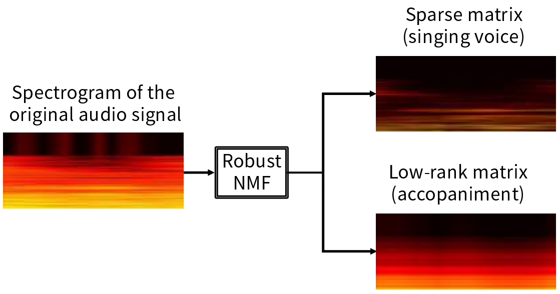

Spectrogram of the original audio signal
Robust NMF
Sparse matrix (singing voice)
Low-rank matrix (accopaniment)

Figure 4. Singing voice separation based on VB-RNMF. The matrix corresponding to an input audio spectrogram is separated into a sparse matrix corresponding to the magnitude spectrogram of singing voices and a low-rank matrix corresponding to the magnitude spectrogram of accompaniment sounds.

3.3 Singing Voice Separation for Music Audio Signals

To separate a musical audio signal specified by the user into singing voices and accompaniment sounds, we propose an online version of variational Bayesian robust NMF (VB-RNMF) [2]. Although there are many offline methods of singing voice separation [17–20], our system requires real-time separation in order to conceal the processing time of the singing voice separation from the user. Figure 4 shows how online VB-RNMF separates a mini-batch spectrogram into a sparse singing-voice spectrogram and a low-rank accompaniment spectrogram.

More specifically, an input spectrogram $\mathbf{Y} = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_T]$ is approximated as the sum of a low-rank spectrogram $\mathbf{L} = [\boldsymbol{l}_1, \ldots, \boldsymbol{l}_T]$ and a sparse spectrogram $\mathbf{S} = [\boldsymbol{s}_1, \ldots, \boldsymbol{s}_T]$:

$$\boldsymbol{y}_t \approx \boldsymbol{l}_t + \boldsymbol{s}_t, \tag{1}$$

where $\mathbf{L}$ is represented by the product of K spectral basis vectors $\mathbf{W} = [\boldsymbol{w}_1, \ldots, \boldsymbol{w}_K]$ and their temporal activation vectors $\mathbf{H} = [\boldsymbol{h}_1, \ldots, \boldsymbol{h}_T]$ as follows:

$$\boldsymbol{y}_t \approx \mathbf{W}\boldsymbol{h}_t + \boldsymbol{s}_t. \tag{2}$$

The tread-off between low-rankness and sparseness is controlled in a Bayesian manner stated below.

The Kullback-Leibler (KL) divergence is used for measuring the approximation error. Since the maximization of the Poisson likelihood (denoted by $\mathcal{P}$) corresponds to the minimization of the KL divergence, the likelihood function is given by

$$p(\mathbf{Y}|\mathbf{W}, \mathbf{H}, \mathbf{S}) = \prod_{f,t} \mathcal{P}\left(y_{ft} \,\middle|\, \sum_k w_{fk} h_{kt} + s_{ft} \right). \tag{3}$$

Since the gamma distribution (denoted by $\mathcal{G}$) is a conjugate prior for the Poisson distribution, gamma priors are put on the basis and activation matrices of the low-rank components as follows:

$$p(\mathbf{W}|\alpha^{wh}, \beta^{wh}) = \prod_{f,k} \mathcal{G}(w_{fk}|\alpha^{wh}, \beta^{wh}), \tag{4}$$

$$p(\mathbf{H}|\alpha^{wh}, \beta^{wh}) = \prod_{k,t} \mathcal{G}(h_{kt}|\alpha^{wh}, \beta^{wh}), \tag{5}$$

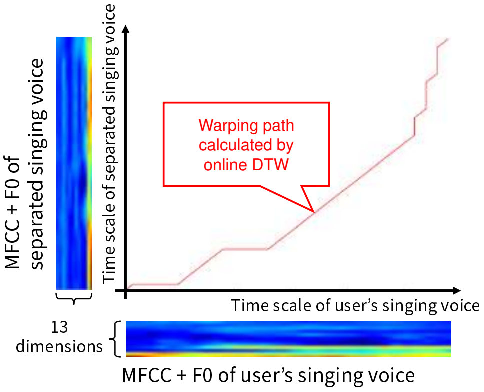


Figure 5. A warping path obtained by online DTW.

where α^{wh} and β^{wh} are represent the shape and rate parameters of the gamma distribution.

To force the sparse components to take nonnegative values, gamma priors with rate parameters given the Jeffreys hyperpriors are put on those components as the following:

$$p(\mathbf{S}|\alpha^s,\boldsymbol{\beta}^s)=\prod_{f,t}\mathcal{G}(s_{ft}|\alpha^s,\beta^s_{ft}), \quad (6)$$

$$p(\beta^s_{ft})\propto(\beta^s_{ft})^{-1}. \quad (7)$$

where α^s represents the hyperparameter of the gamma distribution that controls the sparseness. Using Eqs. (3)–(7), the expected values of $\mathbf{W}$, $\mathbf{H}$, and $\mathbf{S}$ are estimated in a mini-batch style using a VB technique.

3.4 Audio-to-Audio Alignment between Singing Voices

We use an online version of dynamic time warping (DTW) [22] that estimates an optimal warping path between a user's singing voices and original singing voices separated from music audio signals (Figure 5). Since the timbres and F0s of the user's singing voices can significantly differ from those of the original singing voices according to the singing skills of the user, we focus on both MFCCs and F0s of singing voices for calculating a cost matrix in DTW. To estimate the F0s, saliency-based F0 estimation called subharmonic summation [23] is performed.

First MFCCs and F0s are extracted from the mini-batch spectrogram of the separated voice $\boldsymbol{X}=\{\boldsymbol{x}_1,\cdots,\boldsymbol{x}_W\}$ and that of the user's singing voice $\boldsymbol{Y}=\{\boldsymbol{y}_1,\cdots,\boldsymbol{y}_W\}$. Suppose we represent the concatenated vector of MFCCs and F0 extracted from the $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ as $\boldsymbol{x}'_i=\{\boldsymbol{m}^{(x)}_i,f^{(x)}_i\}$ and $\boldsymbol{y}'_i=\{\boldsymbol{m}^{(y)}_i,f^{(y)}_i\}$, $\boldsymbol{X}'=\{\boldsymbol{x}'_1,\cdots\boldsymbol{x}'_W\}$ and $\boldsymbol{Y}'=\{\boldsymbol{y}'_1,\cdots\boldsymbol{y}'_W\}$ are input to the online DTW. In this concatenation, the weight of F0 is smaller than MFCCs. This is because F0 would be much less stable than MFCCs when the user has poor skills. If those mini-bathes are not silent, the MFCCs and F0s are extracted and the cost matrix $D=\{d_{i,j}\}(i=1,\cdots,W;j=1,\cdots,W)$ is updated according to Algorithms 1 and 2 with constraint parameters W, c, and $MaxRunCount$, *i.e.*, a partial row or column of D is calculated as follows:

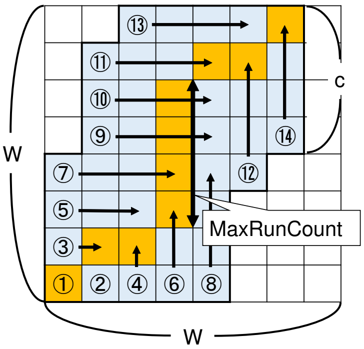


Figure 6. Online DTW with input length $W=8$, search width $c=4$, and path constraint $MaxRunCount=4$. All calculated cells are framed in bold and colored sky blue, and the optimal path is colored orange.

$$d_{i,j}=||\boldsymbol{x}'_i-\boldsymbol{y}'_j||+\min\begin{cases}d_{i,j-1}\\d_{i-1,j}\\d_{i-1,j-1}.\end{cases} \quad (8)$$

The variable s and t in Algorithms 1 and 2 represent the current position in the feature sequences $\boldsymbol{X}'$ and $\boldsymbol{Y}'$ respectively. The online DTW calculates the optimal warping path $L=\{\boldsymbol{o}_1,\cdots,\boldsymbol{o}_l\},\boldsymbol{o}_i=(i_k,j_k)$ $(0\leq i_k\leq i_{k+1}\leq n;0\leq j_k\leq j_{k+1}\leq n)$, using the root mean square for $||\boldsymbol{x}_i-\boldsymbol{y}_j||$ incrementally, without backtraking. (i_k,j_k) means that the frame $\boldsymbol{x}_{i_k}$ corresponds to the frame $\boldsymbol{y}_{j_k}$. Figure 6 shows an example how the cost matrix and the warping path is calculated. Each number in Figure 6 represents the order to calculate the cost matrix. The parameter W is the length of the input mini-batch. If the warping path reaches the W-th row or column, then the calculation stops. If the warping path ends at $(W,k)(k<W)$, the next warping path starts from that point. c restricts the calculation of the cost matrix. At most c successive elements are calculated for each calculation of the cost matrix. $MaxRunCount$ restricts the shape of the warping path. The warping path is incremented at most $MaxRunCount$.

The function $GetInc$ decides which to increment a row, column, or both of the warping path. If the row is incremented from the position (s,t) in the cost matrix, then at most c successive elements from $(s-c,t)$ to (s,t) are calculated. Otherwise, successive elements from $(s,t-c)$ to (s,t) are calculated.

For the system reported here, we set the parameters as $W=300,c=4,MaxRunCount=3$.

3.5 Time Stretching of Accompaniment Sounds

Given the warping path $L=\{\boldsymbol{o}_1,\cdots,\boldsymbol{o}_l\}$, the system calculates a series of stretch rates $R=\{r_1,\cdots,r_W\}$ for each frame of a mini-batch. The stretch rate of the i-th frame, r_i, is given by

$$r_i=\frac{\text{the amount of } i \text{ in } \{i_1,\cdots,i_l\}}{\text{the amount of } i \text{ in } \{j_1,\cdots,j_l\}}. \quad (9)$$

Algorithm 1 The online DTW algorithm

```
$s \leftarrow 1, t \leftarrow 1$, path $\leftarrow (s,t)$, previous $\leftarrow$ None
Calculate $d_{s,t}$ following the eq. 8
while $s < W, t < W$ do
  if GetInc$(s,t) \neq$ Column then
    $s \leftarrow s+1$
    for $k = t-c+1, \cdots, t$ do
      if $k > 0$ then
        Calculate $d_{s,t}$ following the eq. 8
      end if
    end for
  end if
  if GetInc$(s,t) \neq$ Row then
    $t \leftarrow t+1$
    for $k = s-c+1, \cdots, s$ do
      if $k > 0$ then
        Calculate $d_{s,t}$ following the eq. 8
      end if
    end for
  end if
  if GetInc$(s,t) ==$ previous then
    runCount $\leftarrow$ runCount $+1$
  else
    runCount $\leftarrow 1$
  end if
  if GetInc$(s,t) \neq$ Both then
    previous $\leftarrow$ GetInc$(s,t)$
  end if
  path.append($(s,t)$)
end while
```

Then the stretch rate r for the current mini-batch is calculated as the mean of R as $r = \frac{1}{W}\sum_{i=1}^{W} r_i$. r is updated on each iteration of the online DTW. The system finally uses a standard method of time-scale modification called phase vocoder [24] to stretch the mini-batch of the separated accompaniment sounds by a factor of r. The phase vocoder stretches the input sound globally by a factor of r.

4. EVALUATION

We conducted three experiments to evaluate the effectiveness of this proposed system. Quantitatively, we evaluated the efficiency of the system performance and the accuracy of real-time audio-to-audio alignment. We also conducted a subjective experiment.

4.1 Efficiency Evaluation of Singing Voice Separation

To evaluate the efficiency of singing voice separation, we used 100 pieces sampled at 44.1 kHz from the RWC popular music database [25]. Each piece was truncated to 30 seconds from the beginning. Spectrograms were then calculated using short-term Fourier transform (STFT) with the window size of 4096 samples and the hop size of 10 ms. Each spectrogram was then split into 30 300-millisecond mini-batches, which were input to online VB-RNMF.

The average processing time for a 300-millisecond mini-batch was 538.93 ms, and no mini-batches were processed in less than 300 ms. This means that the singing voice separation actually does not work in real time, but it is sufficient to wait a short while before using the system. For greater convenience, the performance of singing voice separation could be improved. One way to achieve this is to process singing voice separation on a graphics processing unit (GPU).

Algorithm 2 The function $GetInc(s,t)$

```
if $s < c$ then
  return Both
end if
if runCount $<$ MaxRunCount then
  if previous $==$ Row then
    return Column
  else
    return Row
  end if
end if
$(x,y) = \arg\min(D(k,l))$, where $k == s$ or $l == t$
if $x < s$ then
  return Row
else if $y < t$ then
  return Column
else
  return Both
end if
```

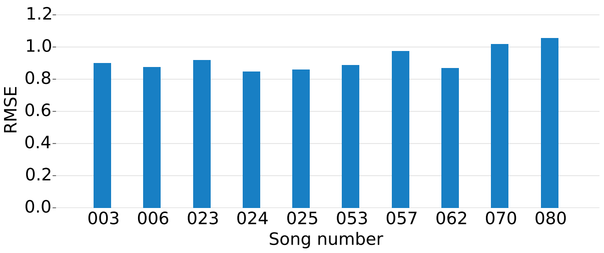


Figure 7. Stretch rate RMSEs measured in the accuracy evaluation. The RMSEs represent how the estimated stretch rates differ from the original rates.

4.2 Accuracy Evaluation of Singing Voice Alignment

To evaluate the accuracy of audio-to-audio alignment, we randomly selected 10 pieces from the database. The singing voices were separated from the 30-second spectrogram of each piece. The phase vocoder [24] was then used to stretch the separated singing voices according to eleven kinds of stretch rates, $r = 0.5, 0.6, \cdots, 1.4, 1.5$. The separated voice and the stretched version of it were input to online DTW, and the stretch rate r' was calculated from the estimated warping path. Then the r' and r were compared. The system uses the separated singing voice and the user's clean voice, but this evaluation uses the separated singing voice and a stretched version of the voice to determine a correct stretch rate.

Figure 7 shows the stretch rate root mean squared errors (RMSEs) between r and r'. The average RMSE over 10 pieces was 0.92 and the standard deviation was 0.068. This indicates that the performance difference of audio-to-audio alignment varied little over different songs, but the align-

	question (1)	question (2)
subject 1	partially yes	partially yes
subject 2	yes	partially yes
subject 3	partially yes	no
subject 4	yes	partially yes

Table 1. The result of the subjective evaluation.

ment accuracy was not very high. This was because the separated singing voices contained musical noise, and the time stretch led to further noise, giving inaccurate MFCCs.

The number of songs used in this evaluation is rather small. We plan further evaluation using more songs.

There are many possibilities for improving the accuracy. First, HMM-based methods would be superior to the DTW-based method. HMM-based methods learn previous inputs unlike DTW-based methods, and this would be useful to improve the accuracy. Second, simultaneous estimation of the alignment and the tempo estimation would improve the accuracy. The result of tempo estimation could help to predict the alignment. An approach for this way of alignment is a method using particle filters [11].

4.3 Subjective Evaluation

Each of four subjects was asked to sing a Japanese popular song after listening to the songs in advance. The songs used for evaluation were an advertising jingle “Hitachi no Ki,” a rock song “Rewrite” by Asian Kung-fu Generation, a popular song “Shonen Jidai” by Inoue Yosui, and a popular song “Kimagure Romantic” by Ikimono Gakari. The subjects were then asked two questions; (1) whether the automatic accompaniment was accurate, and (2) whether the user interface was appropriate.

The responses by the subjects are shown in Table 1. The responses indicate, respectively, that the automatic accompaniment was partially accurate and practical, and the user interface was useful.

The subjects also gave several opinions for the system. First, the accompaniment sounds were low quality and it was not obvious whether the automatic accompaniment was accurate. We first need to evaluate quality of the singing voice separation. An approach for this problem could be to add a mode of playing a click sound according to the current tempo. Second, some of the subjects did not understand what the displayed spectrograms represented. Some explanation should be added for further user-friendliness, or only the stretch rate and F0 trajectories should be displayed.

The number of the test sample used in this subjective evaluation is rather small. We plan further evaluation by more subjects.

5. CONCLUSION

This paper presented a novel adaptive karaoke system that plays back accompaniment sounds separated from music audio signals while adjusting the tempo of those sounds to that of the user’s singing voices. The main components of the system are singing voice separation based on online VB-RNMF and audio-to-audio alignment between singing voices based on online DTW. This system enables a user to expressively sing an arbitrary song by dynamically changing the tempo of the user’s singing voices. The quantitative and subjective experimental results showed the effectiveness of the system.

We plan to improve separation and alignment of singing voices. Using the tempo estimation result would help improvement of the audio-to-audio alignment. Automatic harmonization for users’ singing voices would be an interesting function as a smart karaoke system. Another important research direction is to help users improve their singing skills by analyzing the weak points from the history of the matching results between the user’s and original singing voices.

Acknowledgments

This study was supported by JST OngaCREST and OngaACCEL Projects and JSPS KAKENHI Nos. 26700020, 24220006, 26280089, 16H01744, and 15K16654.

6. REFERENCES

[1] M. Hamasaki *et al.*, “Songrium: Browsing and listening environment for music content creation community,” in *Proc. SMC*, 2015, pp. 23–30.

[2] Y. Bando *et al.*, “Variational bayesian multi-channel robust nmf for human-voice enhancement with a deformable and partially-occluded microphone array,” in *Proc. EUSIPCO*, 2016, pp. 1018–1022.

[3] H. Tachibana *et al.*, “A real-time audio-to-audio karaoke generation system for monaural recordings based on singing voice suppression and key conversion techniques,” *J. IPSJ*, vol. 24, no. 3, pp. 470–482, 2016.

[4] W. Inoue *et al.*, “Adaptive karaoke system: Human singing accompaniment based on speech recognition,” in *Proc. ICMC*, 1994, pp. 70–77.

[5] R. B. Dannenberg, “An on-line algorithm for real-time accompaniment,” in *Proc. ICMC*, 1984, pp. 193–198.

[6] B. Vercoe, “The synthetic performer in the context of live performance,” in *Proc. ICMC*, 1984, pp. 199–200.

[7] C. Raphael, “Automatic segmentation of acoustic musical signals using hidden markov models,” *J. IEEE Trans. on PAMI*, vol. 21, no. 4, pp. 360–370, 1999.

[8] A. Cont, “A coupled duration-focused architecture for realtime music to score alignment,” *J. IEEE Trans. on PAMI*, vol. 32, no. 6, pp. 974–987, 2010.

[9] E. Nakamura *et al.*, “Outer-product hidden markov model and polyphonic midi score following,” *J. New Music Res.*, vol. 43, no. 2, pp. 183–201, 2014.

[10] T. Nakamura *et al.*, “Real-time audio-to-score alignment of music performances containing errors and arbitrary repeats and skips,” *J. IEEE/ACM TASLP*, vol. 24, no. 2, pp. 329–339, 2016.

[11] N. Montecchio *et al.*, “A unified approach to real time audio-to-score and audio-to-audio alignment using sequential montecarlo inference techniques,” in *Proc. ICASSP*, 2011.

[12] R. Gong *et al.*, "Real-time audio-to-score alignment of singing voice based on melody and lyric information," in *Proc. Interspeech*, 2015.

[13] H. Fujihara *et al.*, "LyricSynchronizer: Automatic synchronization system between musical audio signals and lyrics," in *Proc. IEEE Journal of Selected Topics in Signal Processing Conference*, 2011, pp. 1252–1261.

[14] D. Iskandar *et al.*, "Syllabic level automatic synchronization of music signals and text lyrics," in *Proc. ACMMM*, 2006, pp. 659–662.

[15] Y. Wang *et al.*, "LyricAlly: Automatic synchronization of textual lyrics to acoustic music signals," *J. IEEE TASLP*, vol. 16, no. 2, pp. 338–349, 2008.

[16] G. Dzhambazov *et al.*, "Modeling of phoneme durations for alignment between polyphonic audio and lyrics," in *Proc. SMC*, 2015, pp. 281–286.

[17] P.-S. Huang *et al.*, "Singing-voice separation from monaural recordings using robust principal component analysis," in *Proc. IEEE ICASSP*, 2012, pp. 57–60.

[18] Y. Ikemiya *et al.*, "Singing voice separation and vocal F0 estimation based on mutual combination of robust principal component analysis and subharmonic summation," *J. IEEE/ACM TASLP*, vol. 24, no. 11, pp. 2084–2095, 2016.

[19] Z. Rafii *et al.*, "Music/voice separation using the similarity matrix," in *Proc. ISMIR*, 2012, pp. 583–588.

[20] P.-K. Yang *et al.*, "Bayesian singing-voice separation," in *Proc. ISMIR*, 2014, pp. 507–512.

[21] P.-S. Huang *et al.*, "Singing-voice separation from monaural recordings using deep recurrent neural networks," in *Proc. ISMIR*, 2014, pp. 477–482.

[22] S. Dixon, "An on-line time warping algorithm for tracking musical performances," in *Proc. the 19th IJCAI*, 2005, pp. 1727–1728.

[23] D. J. Hermes, "Measurement of pitch by subharmonic summation," *J. ASA*, vol. 83, no. 1, pp. 257–264, 1988.

[24] J. Flanagan *et al.*, "Phase vocoder," *Bell System Technical Journal*, vol. 45, pp. 1493–1509, 1966.

[25] M. Goto *et al.*, "Rwc music database: Popular, classical, and jazz music databases," in *Proc. ISMIR*, 2002, pp. 287–288.

Experimental assessment of descriptive statistics and adaptive methodologies for threshold establishment in onset selection functions

Jose J. Valero-Mas, José M. Iñesta
Pattern Recognition and Artificial Intelligence Group
University of Alicante
{jjvalero,inesta}@dlsi.ua.es

ABSTRACT

Onset detection methods generally work on a two-stage basis: a first step which processes an audio stream and computes a time series depicting the estimated onset positions as local maxima in the function; and a second one which evaluates this time series to select some of the local maxima as onsets, typically with the use of peak picking and thresholding methodologies. Nevertheless, while the former stage has received considerable attention from the community, the second one typically ends up being one of a reduced catalogue of procedures. In this work we focus on this second stage and explore previously unconsidered methods based on descriptive statistics to obtain the threshold function. More precisely, we consider the use of the percentile descriptor as a design parameter and compare it to classic strategies, as for instance the median value. Additionally, a thorough comparison of methodologies considering the temporal evolution of the time series (adaptive techniques) against the use of static threshold values (non-adaptive) is carried out. The results obtained report several interesting conclusions, being the most remarkable two the fact that the percentile descriptor can be considered a competitive possible alternative for this task and that adaptive approaches do not always imply an improvement over static methodologies.

1. INTRODUCTION

Onset detection stands for the process of automatically retrieving the starting points of note events present in audio music signals [1]. This task has been commonly addressed in the Music Information Retrieval (MIR) discipline due to its remarkable relevance in related research fields such as *beat detection* [2], *tempo estimation* [3], or *automatic music transcription* [4].

However, in spite of its conceptual simplicity, this task remains unsolved. While the performance for cases of simple instrumentation and low polyphony degree may be considered sufficient for a number of MIR tasks, in more complex cases onset estimation still constitutes a challenge.

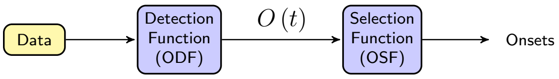


Figure 1: Graphical description of a two-stage onset detection process.

Except for some approaches based on end-to-end systems (e.g, [5]), classic onset estimation schemes base its performance on a process comprising two consecutive steps [6], as shown in Fig. 1. The first stage, usually known as *Onset Detection Function* (ODF), processes the target signal computing a time series whose peaks represent the positions of the estimated onsets, by measuring the change in one or more audio features. Features typically considered are signal energy [7], phase [8], or combinations of them [9].

The second stage, known as *Onset Selection Function* (OSF), evaluates the result from the ODF process and selects the most promising peaks as onsets. Ideally, if the ODF method would perfectly track the onset events in the signal, this stage would not be required. However, as no ODF process is neat, the OSF is required to discriminate between actual onsets and artifacts, thus constituting a key point in the system's performance [10, 11].

Due to its relevance, a number of OSF techniques have been proposed. In general, OSF methods seek for peaks above a certain threshold in the ODF. In this regard, while the maximum condition is typically unaltered, the different techniques differ in the way this threshold is obtained.

The most basic approach consists in manually establishing a static threshold value [12]. This technique does not consider any particularities of the signal for selecting the value, i.e., no prior knowledge of the signal is taken into account. Its value is heuristically set according to previous experimentation. Other more sophisticated approaches consider calculating the mean or the median value of the ODF and setting it as the static threshold value [13, 14].

Extending the premises of the aforementioned methods, adaptive techniques which consider the temporal evolution of the ODF are also used. Instead of obtaining a global static threshold value, these methods include a sliding window for performing the analysis at each point [15]. In this context, the median value of the elements in the sliding window is commonly considered a remarkably robust approach for onset detection [16].

Nevertheless, except for some particular works assessing the influence of the OSF process in the overall performance of the system (e.g., [10]), there is still a need to further study such processes. Thus, in this work we perform an experimental study assessing different ideas which, to the best of our knowledge, no previous work has considered. Particularly, we study the possibility of considering other percentile values (i.e., other statistical tendency measures) different to the median, somehow extending the work by Kauppinen [13] but particularised to onset detection. We also examine the influence of the window size in adaptive methodologies. Eventually, we compare static and adaptive threshold procedures to check their respective capabilities.

The rest of the paper is structured as follows: Section 2 introduces the ODF methods considered for this work; Section 3 explains the different OSF approaches assessed and compared; Section 4 describes the evaluation methodology considered; Section 5 introduces and analyses the results obtained; eventually, Section 6 remarks the conclusions and proposes future work.

2. ONSET DETECTION FUNCTIONS

Two different ODF algorithms were used for our experimentation, which were selected according to their good results reported in the literature: the Semitone Filter Bank (SFB) [7] and the SuperFlux (SuF) [17,18] algorithms. For the rest of the paper we refer to the time series resulting from the ODF process as $O(t)$.

As an energy-based ODF method, SFB analyses the evolution of the magnitude spectrogram with the particularity of considering that harmonic sounds are being processed. A semitone filter bank is applied to each frame window of the magnitude spectrogram, being the different filters centred at each of the semitones marked by the well temperament, and the energy of each band (root mean square value) is retrieved. After that, the first derivative across time is obtained for each single band. As only energy rises may point out onset information, negative outputs are zeroed. Finally, all bands are summed and normalised to obtain the $O(t)$ function.

The SuF method bases its performance on the idea of the Spectral Flux [19] signal descriptor and extends it. Spectral Flux obtains positive deviations of the bin-wise difference between two consecutive frames of the magnitude spectrogram for retrieving the $O(t)$ function. SuF substitutes the difference between consecutive analysis windows by a process of tracking spectral trajectories in the spectrum together with a maximum filtering process that suppresses vibrato articulations as they tend to increase false detections.

Given that the different ODF processes may not span in the same range, we apply the following normalisation process to $O(t)$ so that the resulting function $\hat{O}(t)$ satisfies $\hat{O}(t) \in [0,1]$:

$$\hat{O}(t) = \frac{O(t) - \min\{O(t)\}}{\max\{O(t)\} - \min\{O(t)\}} \tag{1}$$

where $\min\{\cdot\}$ and $\max\{\cdot\}$ retrieve the minimum and maximum values of $O(t)$, respectively.

Finally, it must be mentioned that the analysis parameters of both algorithms have been configured to a window size of 92.9 ms with a 50 % of overlapping factor.

3. ONSET SELECTION FUNCTIONS

In this section we introduce the different OSF methods considered. Given that the contemplated ODF processes depict onset candidates as local maxima in $\hat{O}(t)$, the studied OSF methods search for peaks above a certain threshold value $\theta(t)$ for removing spurious detections.

As introduced, the idea in this work is to assess the following OSF criteria: a) considering percentile values different to the median one; b) assessing the influence of the window size in adaptive methodologies; and c) comparing the performance of static against adaptive methodologies. Mathematically, these criteria can be formalised as in Eq. 2:

$$\theta(t_i) = \mu\left\{\hat{O}(t_{w_i})\right\} + \mathcal{P}^{(n)}\left\{\hat{O}(t_{w_i})\right\} \tag{2}$$

where $t_{w_i} \in \left[t_i - \frac{W}{2}, t_i + \frac{W}{2}\right]$, W denotes the size of the sliding window, and $\mu\{\cdot\}$ and $\mathcal{P}^{(n)}\{\cdot\}$ denote the average and n_{th} percentile value of the sample distribution at issue, respectively.

For assessing the influence of the percentile, 20 values equally spaced in the range $n \in [0, 100]$ were considered. The particular case of $\mathcal{P}^{(50)}$ is equivalent to the median value of the distribution.

In terms of window sizes, 20 values equally space in the range $W \in [0.2, 5]$ s were also considered. Additionally, the value proposed by West and Cox [20] of $W = 1.5$ s was included as a reference.

In order to obviate the adaptive capabilities of the OSF, W was set to the length of the $O(t)$ to assess the static methodologies.

In addition, the case of manually imposing the threshold value $\theta(t_i) = \mathcal{T}$ was considered. For that we establish 20 values equally spaced in the range $\mathcal{T} \in [0, 1]$ as $\hat{O}(t)$ is normalised to that range.

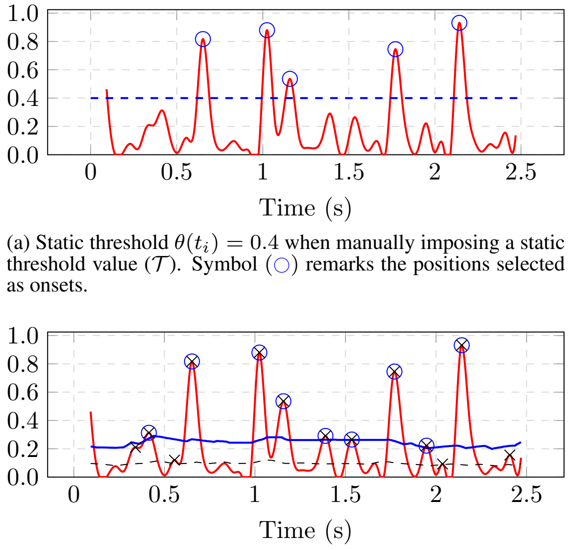
For the rest of the paper, the following notation will be used: μ and $\mathcal{P}$ denote the use of the mean or the percentile, respectively, whereas $\mu + \mathcal{P}$ stands for the sum of both descriptors; adaptive approaches are distinguished from the static ones by incorporating the t as a subindex, showing their temporal dependency (i.e., μ_t, $\mathcal{P}_t$, and $\mu_t + \mathcal{P}_t$ in opposition to μ, $\mathcal{P}$, and $\mu + \mathcal{P}$); $\mathcal{T}$ evinces the manually imposition of the threshold value; lastly, $\mathcal{B}$ is used to denote the case in which no threshold is applied and all local maxima are retrieved as onsets: $\mathcal{B} \equiv \theta(t_i) = 0$.

Finally, Fig. 2 shows some graphical examples of OSF processes applied to a given $\hat{O}(t)$ function.

4. METHODOLOGY

4.1 Corpus

The dataset used for the evaluation is the one proposed by Böck et al. [14]. It comprises 321 monaural recordings sampled at 44.1 kHz covering a wide variety of polyphony degrees and instrumentation.



(a) Static threshold $\theta(t_i) = 0.4$ when manually imposing a static threshold value ($\mathcal{T}$). Symbol (○) remarks the positions selected as onsets.

(b) Adaptive threshold $\theta(t_i)$ using the percentile scheme ($\mathcal{P}_t$): solid and dashed styles represent the 75th and 50th percentiles, respectively, while symbols (○) and (×) depict the positions selected as onsets for each criterion. The sliding window has been set to $W = 1.5\ s$.

Figure 2: Examples of OSF applied to a given $O(t)$.

The total duration of the set is 1 hour and 42 minutes and contains 27,774 onsets. The average duration per file is 19 seconds (the shortest lasts 1 second and the longest one extends up to 3 minutes) with an average figure of 87 onsets per file (minimum of 3 onsets and maximum of 1,132 onsets). For these experiments no partitioning in terms of instrumentation, duration, or polyphony degree was considered as we aim at assessing broader tendencies.

4.2 Performance measurement

Given the impossibility of defining an exact point in time to be the onset of a sound, correctness evaluation requires of a certain tolerance [21]. In this regard, an estimated onset is typically considered to be correct (*True Positive*, TP) if its corresponding ground-truth annotation is within a $\pm 50\ ms$ time lapse of it [1]. *False Positive* errors (FP) occur when the system overestimates onsets and *False Negative* errors (FN) when the system misses elements present in the reference annotations.

Based on the above, we may define the F-measure (F_1) figure of merit as follows:

$$\mathrm{F}_1 = \frac{2 \cdot \mathrm{TP}}{2 \cdot \mathrm{TP} + \mathrm{FP} + \mathrm{FN}} \qquad (3)$$

where $\mathrm{F}_1 \in [0, 1]$, denoting the extreme values 0 and 1 the worst and best performance of the system, respectively.

Other standard metrics such as Precision (P) and Recall (R) have been obviated for simplicity.

5. RESULTS

In this section we introduce and analyse the results obtained for the different experiments proposed. All figures presented in this section show the weighted average of the results obtained for each of the audio files considering the number of onsets each one contains.

Table 1 shows the results obtained for the proposed comparative of static and adaptive threshold methodologies for both ODF processes considered. In this particular case, window size for adaptive methodologies has been fixed to $W = 1.5\ s$ as in West and Cox [20].

An initial remark to point out according to the results obtained is that, in general, the figures obtained with the sliding window methodology do not remarkably differ to the ones obtained in the static one, independently of the ODF process. This can be clearly seen when $\mathcal{P}$ and $\mu + \mathcal{P}$ are respectively compared to $\mathcal{P}_t$ and $\mu_t + \mathcal{P}_t$: all percentile values considered retrieve considerably similar results except for the case when high percentile values are considered, in which the $\mathcal{P}_t$ method shows its superior capabilities. Additionally, while $\mathcal{P}$ approaches show their best performance for percentile values in the range $[60, 75]$, the $\mu + \mathcal{P}$ methods obtain their optimal results in the lower ranges, more precisely around $[0, 30]$.

Regarding the use of the μ methodologies, results obtained for the static and adaptive methodologies did not differ for the SFB function. On the contrary, when considering the SuF function, the adaptive methodology performed slightly worse than the static one. Finally, as these methods do not depend on any external configuration, their performance does not vary with the percentile parameter.

In terms of the $\mathcal{T}$ strategy, its performance matched the static $\mathcal{P}$ and $\mu + \mathcal{P}$ methods in the sense that the performance degrades as the introduced threshold was increased. Nevertheless, this method showed its performance when the threshold value considered lies in the range $[0.10, 0.30]$.

As a general comparison of the methods considered, the μ methods showed a very steady performance paired with high performance results. Also, while $\mu + \mathcal{P}$ methods generally outperform $\mathcal{P}$ strategies in terms of peak performance, the particular case of $\mathcal{P}_t$ approaches showed less dependency on the percentile configuration value.

Given that all previous experimentation has been done considering a fixed window value of $W = 1.5\ s$, we need to study the influence of that parameter in the overall performance of the system. In this regard, Tables 2 and 3 show the results for the $\mathcal{P}_t$ and $\mu_t + \mathcal{P}_t$ methods considering different window sizes for the SFB and SuF processes, respectively. Additionally, Figures 3 and 4 graphically show these results for the $\mathcal{P}_t$ and $\mu_t + \mathcal{P}_t$ methods, respectively.

As the figures obtained for both ODF processes qualitatively show the same trends, we shall analyse them jointly. Checking Tables 2 and 3, results for the $\mathcal{P}_t$ method show that larger windows tend to increase the overall performance of the detection, at least when not considering an extreme percentile value. For instance, let us focus on the $\mathcal{P}_t^{(74)}$ method for the SuF process (Table 3): when considering a window of $W = 0.20\ s$, the performance is set on $\mathrm{F}_1 = 0.67$, which progressively improves as the window size is increased, getting to a score of $\mathrm{F}_1 = 0.73$ when $W = 5.00\ s$.

As commented, this premise is not accomplished when percentile values are set to its possible extremes. For low

Th/Pc	**SFB**							**SuF**						
	$\mathcal{T}$	μ	$\mathcal{P}$	$\mu+\mathcal{P}$	μ_t	$\mathcal{P}_t$	$\mu_t+\mathcal{P}_t$	$\mathcal{T}$	μ	$\mathcal{P}$	$\mu+\mathcal{P}$	μ_t	$\mathcal{P}_t$	$\mu_t+\mathcal{P}_t$
0.00	0.65	**0.73**	0.65	**0.73**	0.72	0.65	0.72	0.64	**0.77**	0.64	**0.77**	0.74	0.64	0.76
0.11	0.71	**0.73**	0.65	0.72	0.72	0.65	0.71	0.76	**0.77**	0.64	**0.77**	0.74	0.64	0.76
0.21	**0.75**	0.73	0.66	0.71	0.72	0.66	0.70	0.65	**0.77**	0.65	**0.77**	0.74	0.64	0.76
0.32	**0.73**	**0.73**	0.66	0.70	0.72	0.66	0.69	0.51	**0.77**	0.66	**0.77**	0.74	0.65	0.76
0.42	0.66	**0.73**	0.68	0.68	0.72	0.67	0.67	0.39	**0.77**	0.68	0.76	0.74	0.67	0.75
0.53	0.56	**0.73**	0.69	0.66	0.72	0.69	0.63	0.29	**0.77**	0.70	0.75	0.74	0.69	0.74
0.63	0.44	**0.73**	0.70	0.62	0.72	0.70	0.58	0.21	**0.77**	0.73	0.73	0.74	0.71	0.71
0.74	0.31	**0.73**	0.70	0.53	0.72	0.70	0.50	0.15	**0.77**	0.74	0.68	0.74	0.72	0.65
0.84	0.19	**0.73**	0.65	0.39	0.72	0.63	0.36	0.10	**0.77**	0.68	0.56	0.74	0.65	0.54
0.95	0.10	**0.73**	0.40	0.13	0.72	0.42	0.12	0.06	**0.77**	0.42	0.29	0.74	0.44	0.27
1.00	0.05	**0.73**	0.05	0.00	0.72	0.27	0.00	0.05	**0.77**	0.05	0.00	0.74	0.29	0.00

Table 1: Results in terms of the F$_1$ score for the dataset considered for the ODF methods introduced. A 1.5-second window has been used for the methods based on sliding window (μ_t, $\mathcal{P}_t$, $\mu_t + \mathcal{P}_t$) based on the work by West and Cox [20]. Bold elements represent the best figures for each ODF method considered. Due to space requirements we have selected the threshold and percentile parameters (Th/Pc) showing the general tendency. In this same regard, both threshold and percentile parameters are expressed in the range $[0, 1]$.

Pc	$W = 0.20$		$W = 0.71$		$W = 1.50$		$W = 2.22$		$W = 2.98$		$W = 3.99$		$W = 5.00$	
	$\mathcal{P}_t$	$\mu_t+\mathcal{P}_t$	$\mathcal{P}_t$	$\mu_t+\mathcal{P}_t$	$\mathcal{P}_t$	$\mu_t+\mathcal{P}_t$	$\mathcal{P}_t$	$\mu_t+\mathcal{P}_t$	$\mathcal{P}_t$	$\mu_t+\mathcal{P}_t$	$\mathcal{P}_t$	$\mu_t+\mathcal{P}_t$	$\mathcal{P}_t$	$\mu_t+\mathcal{P}_t$
0	0.65	0.66	0.65	0.71	0.65	0.72	0.65	0.72	0.65	0.72	0.65	0.72	0.65	**0.73**
11	0.65	0.65	0.65	0.70	0.65	0.71	0.65	0.71	0.65	**0.72**	0.65	**0.72**	0.65	**0.72**
21	0.65	0.63	0.65	0.69	0.66	0.70	0.66	0.70	0.66	**0.71**	0.66	**0.71**	0.66	**0.71**
32	0.65	0.60	0.66	0.68	0.66	0.69	0.66	0.69	0.66	0.69	0.66	**0.70**	0.66	**0.70**
42	0.65	0.56	0.67	0.65	0.67	0.67	0.67	0.67	0.67	**0.68**	0.67	**0.68**	**0.68**	**0.68**
53	0.65	0.49	0.68	0.62	**0.69**	0.64	**0.69**	0.64	**0.69**	0.65	**0.69**	0.65	**0.69**	0.65
63	0.65	0.42	0.69	0.56	**0.70**	0.59	**0.70**	0.59	**0.70**	0.60	**0.70**	0.60	**0.70**	0.60
74	0.67	0.32	0.69	0.48	0.69	0.51	**0.70**	0.51	**0.70**	0.51	**0.70**	0.52	**0.70**	0.52
84	**0.67**	0.12	0.66	0.35	0.64	0.36	0.64	0.36	0.64	0.37	0.64	0.37	0.64	0.37
95	**0.67**	0.00	0.47	0.15	0.45	0.12	0.42	0.12	0.42	0.12	0.41	0.12	0.40	0.12
100	**0.67**	0.00	0.47	0.00	0.23	0.00	0.21	0.00	0.17	0.00	0.13	0.00	0.11	0.00

Table 2: Results in terms of the F$_1$ score for the SFB process. Bold elements represent the best figures obtained for each percentile value considered. Due to space requirements, the most representative percentile parameters (Pc) and window sizes (W) have been selected to show the general tendency.

Pc	$W = 0.20$		$W = 0.71$		$W = 1.50$		$W = 2.22$		$W = 2.98$		$W = 3.99$		$W = 5.00$	
	$\mathcal{P}_t$	$\mu_t+\mathcal{P}_t$	$\mathcal{P}_t$	$\mu_t+\mathcal{P}_t$	$\mathcal{P}_t$	$\mu_t+\mathcal{P}_t$	$\mathcal{P}_t$	$\mu_t+\mathcal{P}_t$	$\mathcal{P}_t$	$\mu_t+\mathcal{P}_t$	$\mathcal{P}_t$	$\mu_t+\mathcal{P}_t$	$\mathcal{P}_t$	$\mu_t+\mathcal{P}_t$
0	0.64	0.72	0.64	0.74	0.64	0.76	0.64	0.76	0.64	**0.77**	0.64	**0.77**	0.64	**0.77**
11	0.64	0.72	0.64	0.75	0.64	0.76	0.64	**0.77**	0.64	**0.77**	0.64	**0.77**	0.64	**0.77**
21	0.64	0.72	0.64	0.75	0.65	0.76	0.65	0.76	0.65	**0.77**	0.65	**0.77**	0.65	**0.77**
32	0.64	0.71	0.65	0.75	0.65	**0.76**	0.65	**0.76**	0.65	**0.76**	0.66	**0.76**	0.66	**0.76**
42	0.64	0.70	0.66	0.74	0.67	0.75	0.67	0.75	0.67	**0.76**	0.67	**0.76**	0.68	**0.76**
53	0.64	0.65	0.68	0.73	0.69	0.74	0.69	0.74	0.70	0.74	0.70	0.74	0.70	**0.75**
63	0.64	0.61	0.70	0.69	0.71	0.71	0.71	0.71	**0.72**	**0.72**	**0.72**	**0.72**	**0.72**	**0.72**
74	0.67	0.55	0.70	0.63	0.72	0.66	0.72	0.66	**0.73**	0.66	**0.73**	0.67	**0.73**	0.67
84	0.67	0.38	**0.68**	0.52	0.67	0.54	0.67	0.54	0.67	0.55	0.67	0.55	0.67	0.55
95	**0.67**	0.00	0.48	0.32	0.47	0.27	0.44	0.27	0.44	0.27	0.43	0.28	0.42	0.28
100	**0.67**	0.00	0.48	0.00	0.24	0.00	0.21	0.00	0.18	0.00	0.15	0.00	0.12	0.00

Table 3: Results in terms of the F$_1$ score for the SuF process. Bold elements represent the best figures obtained for each percentile value considered. Due to space requirements, the most representative percentile parameters (Pc) and window sizes (W) have been selected to show the general tendency.

percentile values, window size seems to be irrelevant to the system. For instance, when considering the SFB method, $\mathcal{P}_t^{(11)}$, the performance measure was always F$_1 = 0.65$ independently of the W value considered.

On the other extreme, very high percentile values suffer a performance decrease as larger window sizes are used. As an example, for the SuF configuration, $\mathcal{P}_t^{(100)}$ with $W = 5.00\ s$ achieved an $\text{F}_1 = 0.12$, while when $W = 0.20\ s$ this figure raised up to $\text{F}_1 = 0.67$.

Results for the $\mu_t + \mathcal{P}_t$ method showed similar tendencies since, when not considering extreme percentile values, the overall performance increased as larger windows were used. Nevertheless, in opposition to the $\mathcal{P}_t$ case, this particular configuration showed an improvement tendency as W is progressively increased for low percentile values.

In general it was observed that, for all W window sizes, the best percentile configurations seemed to be in the range $[60, 70]$ for the $\mathcal{P}_t$ approach and in the range $[0, 20]$ for the $\mu_t + \mathcal{P}_t$ case. This fact somehow confirms the hypothesis suggesting that the median value may not always be the best percentile to consider.

Checking now the results considering Figs. 3 and 4, some additional remarks, which are more difficult to check in the aforementioned tables, may be pointed out.

The first one of them is that the selection of the proper parameters of windows size and percentile factor is crucial. For both $\mathcal{P}_t$ and $\mu_t + \mathcal{P}_t$ methods, there is a *turning point* in which the performance degrades to values lower than the considered baseline $\mathcal{B}$ (no OSF applied). For the $\mathcal{P}_t$ method there is a clear point for this change in tendency around the 85th percentile for any window size considered. However, for the $\mu_t + \mathcal{P}_t$ approach there was not a unique point but a range, which remarkably varied depending on the type of ODF and window size considered.

Another point to highlight is that the static methodologies ($\mathcal{P}$ and $\mu + \mathcal{P}$) consistently define the upper bound in performance before the so-called turning point. This fact somehow confirms the initial idea of using large windows (on the limit, one single window considering the whole $O(t)$ function) in opposition to small windows.

The results obtained when considering window sizes in the range $[0, 1]$ seconds, which include the performance of the considered reference window of $W = 1.5\ s$, achieved results similar to the obtained upper bound. The other considered window sizes showed a remarkable variability in the performance, ranging from achieving figures similar to the upper bound to figures close to the baseline $\mathcal{B}$.

As a general summary for all the experimentation performed, we may conclude that adaptive OSF methodologies may be avoided as static approaches obtain similar results with less computational cost. Particularly, in our experiments, methods considering percentile ($\mathcal{P}$) or mean and percentile ($\mu + \mathcal{P}$) reported the upper bounds in performance.

Nevertheless, the particular percentile value used remarkably affects the performance. For $\mathcal{P}$, the best results seem to be obtained when the percentile parameter was set in the range [60,70]. For $\mu + \mathcal{P}$ the best figures were obtained when this parameter was set to a very low value.

Finally, the commonly considered median value for the OSF did not report the best results in our experiments. These results point out that the median statistical descriptor may not always be the most appropriate to be used, being necessary to be tuned for each particular dataset.

5.1 Statistical significance analysis

In order to perform a rigorous analysis of the results obtained and derive strong conclusions out of them, we now perform a set of statistical tests. Specifically, these analyses were performed with the non-parametric Wilcoxon rank-sum and Friedman tests [22], which avoid any assumption about the distribution followed by the figures obtained. The former method is helpful for comparisons of different distributions in a pairwise fashion while the latter one generalises this pairwise comparison to a generic number of distributions.

In this work the Wilcoxon rank-sum test was applied to assess whether there are significant differences among all the methods proposed using the results in Table 1. The single scores obtained for each Th/Pc constitute a sample of the distribution for the OSF method to be tested. The results obtained when considering a significance level of $p < 0.05$ can be checked in Table 4.

Attending to the results of the Wilcoxon test, an initial point to comment is that method $\mathcal{T}$ the less robust of all considered due to its significantly lower performance in all cases. Oppositely, method μ can be considered the best strategy as it outperforms all other strategies. The adaptive equivalent to this technique, μ_t, achieves similar performances to the static one except for the case of SuF, in which it does not outperform the rest of the methods as consistently as in the static case. Methods $\mathcal{P}$ and $\mathcal{P}_t$ show an intermediate performance among the previously commented extremes. Their results are not competitive with any of the μ or μ_t strategies. These strategies are generally tied with methods $\mu+\mathcal{P}$ and $\mu_t+\mathcal{P}_t$ as they typically show significantly similar performances with punctual cases in which one of those methods improves the rest.

We now consider the statistical analysis of influence of the window size (W) and the percentile values (Pc) on the overall performance. Given that now we have two variables to be evaluated, we consider the use of the Friedman test. The idea with this experiment is to assess whether variations in these variables report statistically significant difference in the F_1 score, so we assume that the null hypothesis to be rejected is that no difference should be appreciated. Note that this type of analysis does not report whether a particular strategy performs better than another one but simply if the differences when using different parameters are relevant.

For performing this experiment we consider the data in Tables 2 and 3 as they contain all the information required for the analysis. In this regard Tables 5 and 6 show the p significance values obtained when measuring the influence of W and Pc, respectively.

Attending to the results obtained, we can check the remarkable influence of the W parameter in the overall results as all cases report very low p scores which reject the null hypothesis. The only exception is the $\mathcal{P}_t$ in the SFB case in which p is not that remarkably low, but still would reject the null hypothesis considering a typical significance threshold of $p < 0.05$.

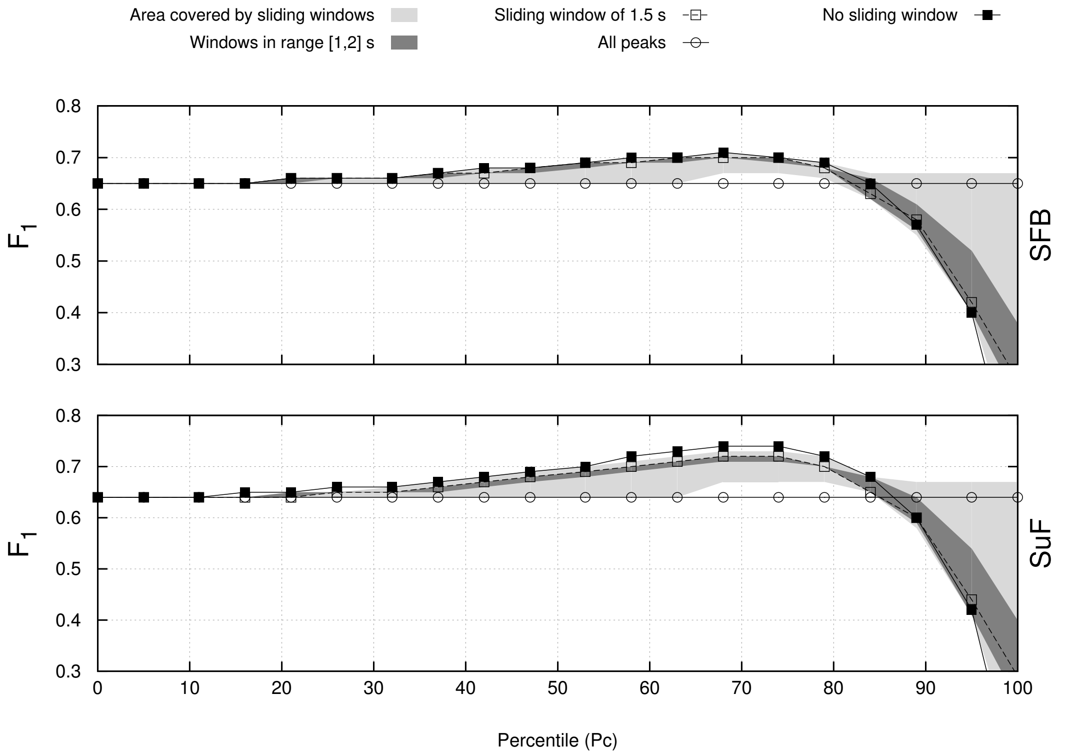


Figure 3: Evolution of the F_1 score when considering the $\mathcal{P}_t$ strategy for the dataset proposed.

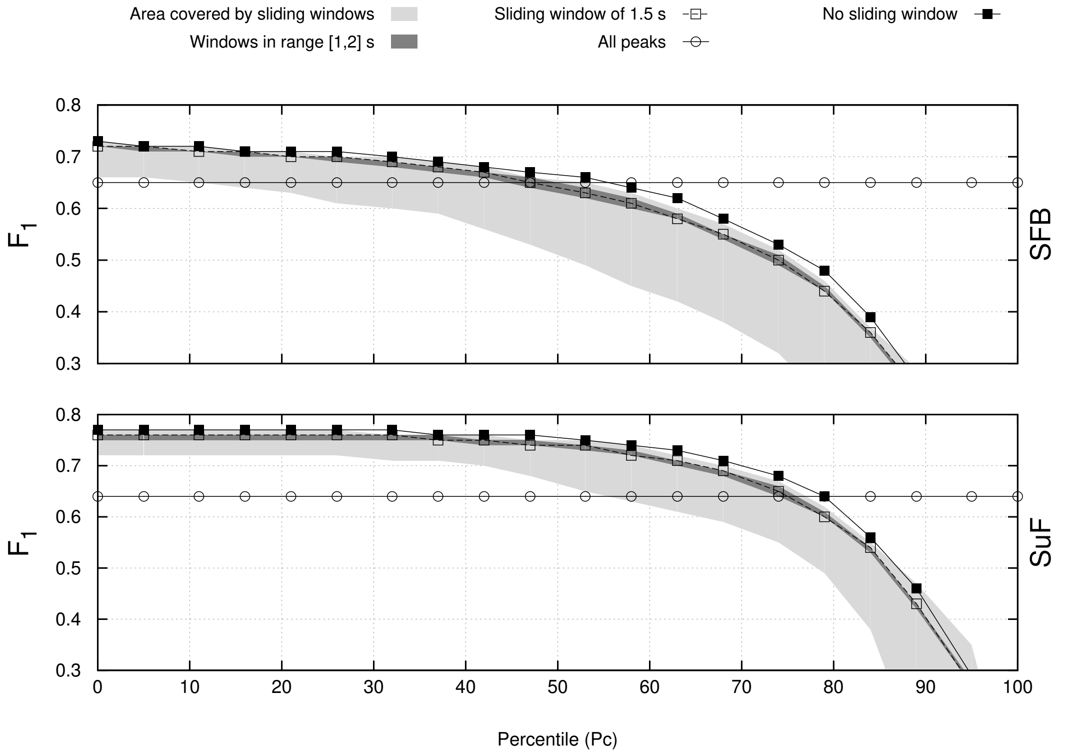


Figure 4: Evolution of the F_1 score when considering the $\mu_t + \mathcal{P}_t$ strategy for the dataset proposed.

In terms of the percentile value (Pc), the p significance values obtained clearly show the importance of this parameter in the design of the experiment. This points out the clear need to consider this as another parameter in the design of onset detection systems.

Finally, note that this statistical analysis confirms the initial conclusions depicted previously: static approaches are significantly competitive when compared to adaptive methods; window sizes for adaptive method remarkably influence the performance of the system; when considering

Method	**SFB**							**SuF**						
	$\mathcal{T}$	μ	$\mathcal{P}$	$\mu+\mathcal{P}$	μ_t	$\mathcal{P}_t$	$\mu_t+\mathcal{P}_t$	$\mathcal{T}$	μ	$\mathcal{P}$	$\mu+\mathcal{P}$	μ_t	$\mathcal{P}_t$	$\mu_t+\mathcal{P}_t$
$\mathcal{T}$	–	$<$	$<$	$<$	$<$	$<$	$<$	–	$<$	$<$	$<$	$<$	$<$	$<$
μ	$>$	–	$>$	$>$	$>$	$>$	$>$	$>$	–	$>$	$>$	$>$	$>$	$>$
$\mathcal{P}$	$>$	$<$	–	$=$	$<$	$=$	$=$	$>$	$<$	–	$=$	$<$	$>$	$=$
$\mu+\mathcal{P}$	$>$	$<$	$=$	–	$<$	$=$	$>$	$>$	$<$	$=$	–	$=$	$=$	$>$
μ_t	$>$	$<$	$>$	$>$	–	$>$	$>$	$>$	$<$	$>$	$=$	–	$>$	$=$
$\mathcal{P}_t$	$>$	$<$	$=$	$=$	$<$	–	$>$	$>$	$<$	$<$	$=$	$<$	–	$=$
$\mu_t+\mathcal{P}_t$	$>$	$<$	$=$	$<$	$<$	$<$	–	$>$	$<$	$=$	$<$	$=$	$=$	–

Table 4: Results obtained for the Wilcoxon rank-sum test to the comparison results depicted in Table 1. Symbols ($>$), ($<$), and ($=$) state that the onset detection capability of the method in the row is significantly higher, lower, or not different to the method in the column. Symbol (–) depicts that the comparison is obviated. A significance level of $p < 0.05$ has been considered.

Method	**SFB**	**SuF**
$\mathcal{P}_t$	0.03209	$4.917 \cdot 10^{-9}$
$\mu_t+\mathcal{P}_t$	$< 2.2 \cdot 10^{-16}$	$< 2.2 \cdot 10^{-16}$

Table 5: Statistical significance results of the Friedman test when measuring the influence of the window size (W) in the overall performance.

Method	**SFB**	**SuF**
$\mathcal{P}_t$	$< 2.2 \cdot 10^{-16}$	$< 2.2 \cdot 10^{-16}$
$\mu_t+\mathcal{P}_t$	$< 2.2 \cdot 10^{-16}$	$< 2.2 \cdot 10^{-16}$

Table 6: Statistical significance results of the Friedman test when measuring the influence of the percentile (Pc) in the overall performance.

OSF methods relying on statistical descriptions, the percentile should be considered as another design parameter due to its relevance in the results.

6. CONCLUSIONS

Onset detection, the retrieval of the starting points of note events in audio music signals, has been largely studied because of its remarkable usefulness in the Music Information Retrieval field. These schemes usually work on a two-stage basis: the first one derives a time series out of the initial signal whose peaks represent the candidate onset positions; the second assesses this function by setting a threshold to minimise the errors and maximise the overall performance. While the first of such processes has received significant attention from the community, the second stage has not been that deeply studied.

In this work we have studied the actual influence of a proper design of this second stage in the overall performance. More precisely we addressed three main points: measuring the benefits of static and adaptive threshold techniques, the use of different statistical descriptions for setting this threshold, and the influence of the considered window size in adaptive methodologies.

Our experiments show that static methodologies are clearly competitive when compared to adaptive methodologies. Additionally, our results report the remarkable influence that the different statistical descriptions and window sizes considered have on the overall performance, being thus important parameters to be considered in onset detection systems.

Future work considers extending this study to other descriptive statistic measures such as central tendency descriptors (e.g., harmonic and weighted means) or shape-based features (e.g., kurtosis and skewness), and considering the possibility of including specific weights for each of them. Also, the exploration of different machine learning methods to avoid use of hand-crafted rules by automatically inferring the optimal threshold curve are also considered as a line to pursue. Finally, end-to-end systems based on deep learning is also a line to further pursue.

Acknowledgments

This research work is partially supported by Vicerrectorado de Investigación, Desarrollo e Innovación de la Universidad de Alicante through FPU program (UAFPU2014--5883) and the Spanish Ministerio de Economía y Competitividad through project TIMuL (No. TIN2013–48152–C2–1–R, supported by EU FEDER funds). Authors would also like to thank Sebastian Böck for kindly sharing the onset dataset.

7. REFERENCES

[1] J. P. Bello, L. Daudet, S. A. Abdallah, C. Duxbury, M. E. Davies, and M. B. Sandler, "A Tutorial on Onset Detection in Music Signals," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 13, no. 5, pp. 1035–1047, 2005.

[2] D. P. W. Ellis, "Beat tracking by dynamic programming," *Journal of New Music Research*, vol. 36, no. 1, pp. 51–60, 2007.

[3] M. Alonso, G. Richard, and B. David, "Tempo Estimation for Audio Recordings," *Journal of New Music Research*, vol. 36, no. 1, pp. 17–25, 2007.

[4] E. Benetos and S. Dixon, "Polyphonic music transcription using note onset and offset detection," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP*, Prague, Czech Republic, 2011, pp. 37–40.

[5] J. Schlüter and S. Böck, "Improved musical onset detection with convolutional neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 6979–6983.

[6] R. Zhou, M. Mattavelli, and G. Zoia, "Music Onset Detection Based on Resonator Time Frequency Image," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 8, pp. 1685–1695, 2008.

[7] A. Pertusa, A. Klapuri, and J. M. Iñesta, "Recognition of Note Onsets in Digital Music Using Semitone Bands," in *Proceedings of the 10th Iberoamerican Congress on Pattern Recognition, CIARP*, Havana, Cuba, 2005, pp. 869–879.

[8] J. P. Bello and M. B. Sandler, "Phase-based note onset detection for music signals," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP*, Hong Kong, Hong Kong, 2003, pp. 49–52.

[9] E. Benetos and Y. Stylianou, "Auditory Spectrum-Based Pitched Instrument Onset Detection," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 8, pp. 1968–1977, 2010.

[10] C. Rosão, R. Ribeiro, and D. Martins de Matos, "Influence of peak picking methods on onset detection," in *Proceedings of the 13th International Society for Music Information Retrieval Conference, ISMIR*, Porto, Portugal, 2012, pp. 517–522.

[11] S. Dixon, "Onset detection revisited," in *Proceedings of the 9th International Conference on Digital Audio Effects, DAFx-06*, Montreal, Canada, 2006, pp. 133–137.

[12] A. Klapuri, "Sound onset detection by applying psychoacoustic knowledge," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP*, vol. 6, Phoenix, USA, 1999, pp. 3089–3092.

[13] I. Kauppinen, "Methods for detecting impulsive noise in speech and audio signals," in *Proceedings of the 14th International Conference on Digital Signal Processing, DSP*, vol. 2, 2002, pp. 967–970.

[14] S. Böck, F. Krebs, and M. Schedl, "Evaluating the Online Capabilities of Onset Detection Methods," in *Proceedings of the 13th International Society for Music Information Retrieval Conference, ISMIR*, Porto, Portugal, 2012, pp. 49–54.

[15] C. Duxbury, J. P. Bello, M. Davies, and M. Sandler, "Complex Domain Onset Detection for Musical Signals," in *Proceedings of the 6th International Conference on Digital Audio Effects, DAFx-03*, London, UK, 2003, pp. 90–93.

[16] J. P. Bello, C. Duxbury, M. Davies, and M. B. Sandler, "On the use of phase and energy for musical onset detection in the complex domain," *IEEE Signal Processing Letters*, vol. 11, pp. 553–556, 2004.

[17] S. Böck and G. Widmer, "Maximum Filter Vibrato Suppression for Onset Detection," in *Proceedings of the 16th International Conference on Digital Audio Effects (DAFx-13)*, Maynooth, Ireland, September 2013, pp. 55–61.

[18] ——, "Local Group Delay based Vibrato and Tremolo Suppression for Onset Detection," in *Proceedings of the 13th International Society for Music Information Retrieval Conference (ISMIR)*, Curitiba, Brazil, November 2013, pp. 589–594.

[19] P. Masri, "Computer Modeling of Sound for Transformation and Synthesis of Musical Signals," PhD Thesis, University of Bristol, UK, 1996.

[20] K. West and S. Cox, "Finding An Optimal Segmentation for Audio Genre Classification," in *Proceedings of the 6th International Conference on Music Information Retrieval, ISMIR*, London, UK, 2005, pp. 680–685.

[21] P. Leveau and L. Daudet, "Methodology and tools for the evaluation of automatic onset detection algorithms in music," in *Proceeding of the 5th International Symposium on Music Information Retrieval, ISMIR*, Barcelona, Spain, 2004, pp. 72–75.

[22] J. Demsar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

Robinflock: a Polyphonic Algorithmic Composer for Interactive Scenarios with Children

Raul Masu
interaction Lab,
University of Trento
raul.masu@unitn.it

Andrea Conci
interaction Lab,
University of Trento
Andrea.Conci.1@ unitn.it

Cristina Core
interaction Lab,
University of Trento
cristina.core@ unitn.it

Antonella de Angeli
interaction Lab,
University of Trento
antonella.deangeli@unitn.it

Fabio Morreale
Centre for Digital Music, EECS
Queen Mary University of London
f.morreale@ qmul.ac.uk

ABSTRACT

The purpose of this paper is to present Robinflock, an algorithmic system for automatic polyphonic music generation to be used with interactive systems targeted at children. The system allows real time interaction with the generated music. In particular, a number of parameters can be independently manipulated for each melody line. The design of the algorithm was informed by the specific needs of the targeted scenario. We discuss how the specific needs of the polyphony with children influenced the development choices. Robinflock was tested in a field study in a local kindergarten involving 27 children over a period of seven months.

1. INTRODUCTION

Polyphonic music is widespread in western music tradition. It consists of at least two melodic lines that are simultaneously performed and that produce coherent harmonies when overlap. Polyphonic music originated with vocal music, grown during the Renaissance [22], and has been fundamental for classical composers (e.g. Bach, Haydn, and Mozart), for modern composers (e.g. Schoenberg and Stockhausen), and for music studies [31].

Performing polyphonic music is particularly demanding as it requires musicians to master score reading and the ability to play with other performers. The difficulties of performing polyphonic music are even more demanding for young children due to their limited musical experience. Despite these difficulties, children seem to have an innate predilection for polyphony [33].

We argue that an algorithmic solution can increase children confidence with polyphonic music. In particular, an algorithmic composition system that generates polyphonic music in real time can be applied in interactive situations to familiarise children with polyphony. We propose to delegate part of the complexities of performing polyphonic music to the computer, enabling children to directly manipulate different lines by interacting with a number of parameters.

The contribution of this paper is the proposition of a real time algorithmic composer of polyphonic music. Indeed, to the best of our knowledge, related studies in algorithmically generated polyphonic music never address interactive purposes. As a consequence, the development of an algorithm that generates polyphonic music in real time was not strictly necessary. Previous algorithmic systems for polyphonic music generation exist; however, they have been developed from a computational linguistic perspective, aiming at generating polyphonic music that avoid composition and counterpoint errors. The objective of these studies was to imitate specific styles [8] or to apply compositional rules derived from music theory [2, 10, 32]. These algorithms successfully generated good quality counterpoint compositions, but their application domain was limited to off-line non-interactive music.

The novelty of the Robinflock lies in the possibility to independently manipulate each musical line on several musical parameters in real time. Robinflock was developed in the context of Child Orchestra [9], a research project aimed at exploring new solutions to use technology in context of music making for children. In this paper, we describe the technical implementation of the algorithm underling how the technical choices are grounded on specific requirements: design solutions were taken to meet a number of requirements related to the specific interactive scenario of polyphonic music for children. The architecture is based on Robin, an algorithm developed by some of the author that generates piano music for interactive purposes [24, 26].

Robinflock was adopted in the Child Orchestra field study, which involved 27 kindergarten children over a period of seven months [9]. The result of this field study suggested that the algorithm is particularly effective to help children to familiarise with polyphonic music.

The remainder of this paper is organized as follows. Section 2 presents related work in algorithmic composition and in music for children. Section 3 describes in details the architecture of Robinflock. Section 4 introduces the field work in which the algorithm was used and presents the results. We conclude this paper with discussions and considerations on future work.

2. RELATED WORK

The work presented in this paper is mainly based on related literature in algorithmic composition. We also present the main child-related musical theories on which we grounded the choices at the basis of the interactive system.

2.1 Algorithmic composition

Algorithmic composition has been widely explored for both research and artistic purposes since the beginning of computer music [17] (see [11, 28] for comprehensive surveys). Given the scope of this paper, we here mainly focus on real time algorithmic systems and systems that generate polyphonic music.

2.1.1 Algorithmically generated polyphonic music

Researchers in algorithmic composition explored polyphony and counterpoint, i.e. the set of rules to compose polyphonic music, mainly with a generative linguistic perspective, mostly with the objective of exploring the potential of computers to imitate specific styles.

David Cope's pioneering work in *Experiment with Music Intelligent* (EMI) approached counterpoint to compute different voices in a specific music style, for instance in a Bach-style Chorale. EMI works by analysing, deconstructing, and then recombining elements of existing music. EMI obtained great results in emulating the style of different composers [8].

Other studies explored counterpoint species (see [19] for a definition of species.) For instance, Aguilera developed an algorithm that generates first specie counterpoint using probabilistic logic [2]. Other systems explored the possibilities to compute more complex polyphonic structures adopting fifth specie counterpoint. For instance, Polito, Daida, and Bersano-Begey [32] developed a genetic algorithm that can compose fifth specie counterpoint based on an existing cantus firmus. Their system adopted a fitness function based on rules proposed by Fux for the counterpoint species. Similarly, Herremans and Soorensen [12] developed Optimuse, a neighbourhood search that generates fifth species counterpoint fragments. Another approach was proposed by Martín-Caro [23] who developed a system to compute two-voices fifth species counterpoint using a first-order Markov chain.

Other systems were developed specifically to algorithmically compose four-part counterpoint music. Among these systems, Phon-Amnuaisu developed a genetic algorithm [30] based on a subset of the four-part rules as a fitness function and Donnelly [10] presented an algorithm based on melodic, harmonic, and rhythmic rules.

All the presented systems achieved a good quality imitation of renaissance and baroque counterpoint. But, as interactivity was behind the scope of these researches, none of them allowed a manipulation of the different line in real time.

2.1.2 Real time interactive algorithmic systems

Real time algorithms have been largely explored for performative purposes, especially to create dialogues with human musicians. Most of these systems adopted a multi-agent architecture. For instance, Rowe's *Cypher* [34] is a real time algorithmic music system designed to improvise with human performers. The system is composed by two main agents, a listener and a player and coordinates a number of small modules such as chord analyser, beat tracking, phrase grouping, and other compositional modules that generate the new musical output. A similar approach was adopted in Lewis' *Voyager*. Indeed, in Lewis' system, the control on the music is shared among 16 "players" [21]. Another real time algorithmic system based on a multi-agent architecture is the *Free Improvisation Simulation* by Nicholas Collins [7].

The above described systems have been mainly developed with performative purposes. Interactive algorithmic composition systems have also been developed with different objectives. The Roboser, for instance, is an algorithmic composition system developed to use real-world behaving as input source to an algorithmic composition system [40]. Another example is Music Blox, which explores the use of generative algorithm in the context of interactivity [12].

With respect to the role of a specific style in interactive algorithms, Aspromallis and Gold recently presented a study on a method for interactive real time music generation that preserves traditional musical genres [3].

All these systems generate a quite large variety of musical genre for interactivity, but the adoption of polyphonic music in this contest is currently overlooked.

2.2 Music and children

In this section, we introduce relevant aspect of music pedagogy and some musical interactive systems for children.

2.2.1 Music theories for children

Since the beginning of the last century, a number of theories have been proposed to discuss musical activities for children [12, 14, 18, 37].

Music Learning Theory by Gordon, one of the most relevant theories for young children, underlines the importance of learning music by ear [14]. Learning by ear implies the development of the listening abilities, which are fundamental also for spatio-temporal reasoning [16]. Listening abilities can be enhanced by physical interaction and active participation of the children, as active participation engages children's attention and memory. Physical interaction and active participation increase children's attention as they are required to respond to the music by changing their body movements [36]. In addition, body movements could help to memorise rhythms [18]. The awareness of rhythmic patterns in music is promoted as it helps the development the control of body language and perceptual skills [14, 18]. A general consensus also suggests that to maximize musical experience that target children should be both complex and musically coherent [14].

2.2.2 Interactive music systems for children

In this paragraph, we introduce musical tools and interactive systems aimed at providing musical experiences to a young population. Some systems were developed to allow children to manipulate musical tracks through their movements. An example of body interaction for children aged between 7 and 10 years is offered by Antle and colleagues

[4]. Their system, which was designed with the help of four music experts, proposed a metaphor based on body movements. The movement of the children were captured by a tracking system and mapped into a percussive sound output, which varied according to the children's movements. Specifically, *tempo* was mapped to the speed of the children, *volume* to the intensity of the activity, and *pitch* to the proximity among children. The system did not control harmonic and melodic features, and consequently did not require a complex algorithmic system to deal with such elements. Another example is the *Continuator*, that is particularly relevant for the scope of this paper as it involves algorithmic elements [29, 1]. The *Continuator*, takes as input the music performed on the keyboard and plays back coherent excerpts that mirror performer's play. The system facilitated piano improvisation by automatically generating musical replies. The system resulted to be effective in promoting flow state for children [1], but does not have any aim in acculturating children in any particular musical technicality.

3. SYSTEM ARCHITECTURE

The algorithmic composer was developed following a number of musical needs that emerged during a preliminary design phase. The design phase, whose details are discussed in a related publication [34], involved experts in the domain of music theory, music teaching, and pedagogy, who participated in four focus groups and one workshop. We present here the main musical needs that emerged is the design phase.

The generated music should be varied, coherent, and compliant with compositional rules, thus aiming to provide children with a valid stimulus for learning by ear [14].

Moreover, the system is required to allow multiple children to interact with it at the same time. Thus, each child needs to have control over a specific voice of the music. In particular, low-level parameters should be directly accessible to actively stimulate the children and foster participation. The parameters individuated during the design phase were volume, speed, and articulation.

These interactive and musical requirements can be met by using an algorithmic solution insofar as it allows to generate and manipulate the different lines in real time. This would provide each child with the control over his own melodic line, thus allowing children to modify the music without reciprocal interference. Furthermore, as opposed to a solution based on triggering pre-recorded melodies, algorithmic composition guarantees continuous access and control to structural parameters. Finally, an algorithmic solution can guarantee a virtually infinite number of compositions but, at the same time, restrain the musical outputs to correct melodic and harmonic rules.

As highlighted in the related work section, most of the systems that algorithmically generate polyphonic music are not designed for interactive purposes [2, 10, 12, 23, 30, 32]. We therefore decided to base the architecture of the algorithm on Robin, an algorithm previously developed by some of the authors [24, 26], which follows a rule-based approach to manipulate a number of musical parameters in real-time (i.e. speed, articulation, melody direction, timbre, volume, consonance, and mode). Robin

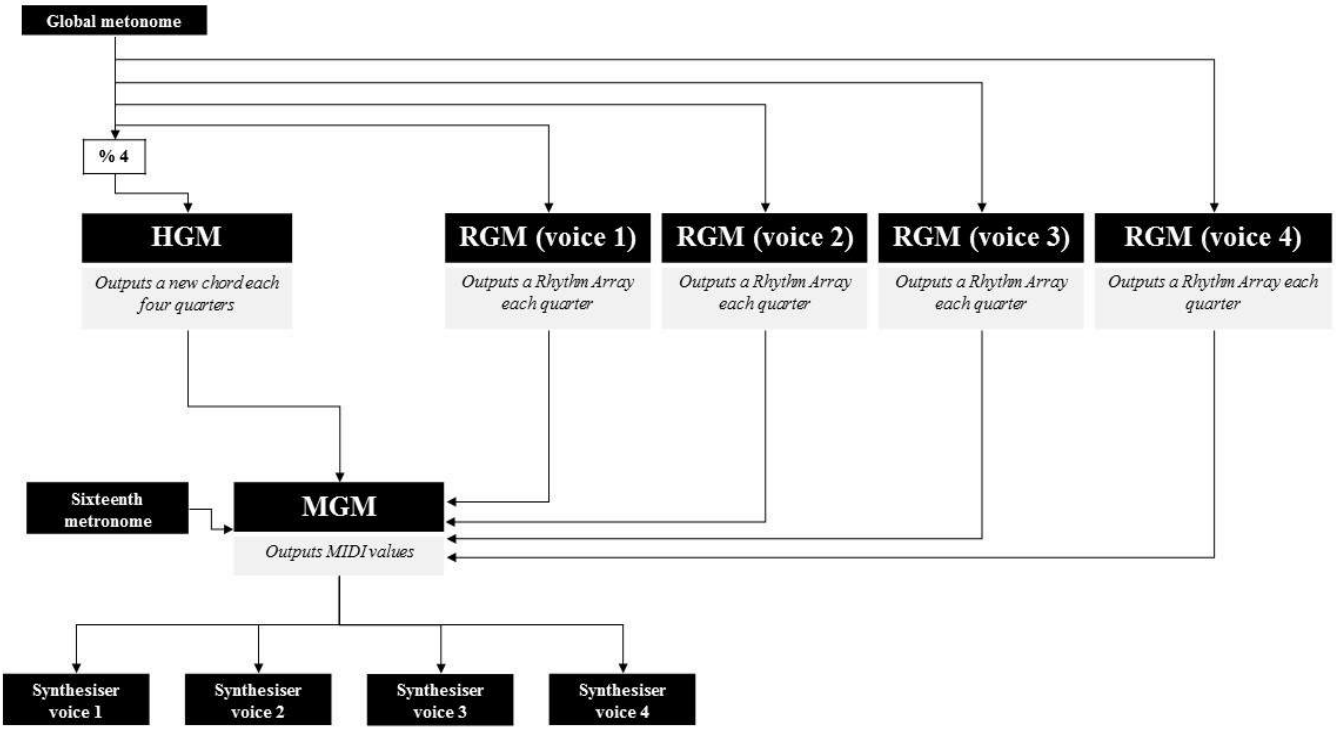


Figure 1. Scheme of the architecture of the algorithm.

controlled a single voice with accompaniment generated by using some piano clichés of classical music, and was adopted in several interactive contexts [25, 27]. However, as each child needs to control a different voice, the algorithm had to be extended to support multiple voices. To this end, we implemented an algorithm that allows independent manipulation on each line of polyphonic music. Robinflock was developed with a rule based approach.

Similarly to most of the interactive systems previously described [7, 21, 34], the structure of Robinflock is modular. The algorithm is composed of three modules that separately deal with: i) *harmony, rhythm, and melody.* The Harmony Generation Module (HGM) follows traditional harmonic rules [31, 35] implemented with a first order Markov chain. The Rhythm Generation Module (RGM) is composed of four modules that compute the rhythm of each line. The rhythm is computed each quarter while the harmony is computed each four quarter. A global metronome synchronises the two modules. The Melodies Generation Module (MGM) fills all the four rhythms with notes that are coherent to the current chord to avoid counterpoint errors like parallel fifths and octaves [17]. The generated music is performed with four FM synthesisers, one for each line. The overall architecture of Robinflock can be seen in Figure 1.

Robinflock is developed in Max-MSP, and each module is implemented with JavaScript, using the Max-JS object, with the only exception of the rhythm module, which is composed of four JS objects, one for each line.

To ensure time synchronisation of the four voices, a metronome is set to the minimum metric unit (the sixteenth) and each voice is encoded using arrays of sixties. All the four arrays are generated by the MGM, which combines the four rhythm arrays generated by the RGMs with the current harmony. A metronome scans the four arrays at the same time: each cell of the array (sixteenth note) contains a MIDI pitch value in case a new note has to be performed or a -1 flag if a new note is not required. In the latter case, the previous note continues to play. These MIDI values are sent to the four synthesisers, that are also implemented in Max-MSP.

3.1 Harmony Generation Module

Traditionally in tonal/modal music, harmony is organized on the basis of chord progressions and cadences. One common approach to manage harmony is that of Generative Grammar, which can compute good quality chord progressions but it is based on time-span reduction of the musical structure, and requires a knowledge *a priori* on the overall structure [38]. Following the design requirements, we needed a high level of flexibility in real time. Therefore, we computed harmony as a stream of chords that are not a priori generated. Each chord is defined as a scale degree and computed as a state of a Markov chain. Since the historical works of Xenakis, Markov chains have been adopted in algorithmic composition [39] to manage temporal successions of musical events. Concerning harmony, the transition probabilities between successive chords have already been modelled as a Markov process [33]. Chords transition data can be extracted by the analysis of existing music [8], surveying of music theory [33], or following personal composing aesthetic [39].

To guarantee a high degree of music adaptability, chord and degrees' correlation does not depend on previous states of the system: a first-order Markov process determines the harmonic progression as a continuous stream of chords. We implemented the matrix following a rule-based approach deriving the transition probabilities from harmony manuals (Table 1) [31, 35].

As the music was required to offer a variety of stimuli, Robinflock was developed to compute music in all the major and minor keys, and in all the Gregorian modalities.

To cope with this requirement, the system computes the harmony progression applying the scale degree to the desired tonality or modality. For example, a first degree in C major key corresponds to the C-E-G chord while the same degree in G minor key corresponds to the G-Bb-D chord.

	I	**II**	**III**	**IV**	**V**	**VI**	**VII**
I	0	0,05	0,07	0,35	0,35	0,08	0,1
II	0,05	0	0,05	0,15	0,65	0,2	0
III	0	0,07	0	0,2	0,8	0,65	0
IV	0,15	0,15	0,05	0	0,6	0,05	0
V	0,64	0,05	0,05	0,13	0	0,13	0
VI	0,06	0,35	0,12	0,12	0,35	0	0
VII	1	0	0	0	0	0	0

Table 1. The Markov matrix in the Harmony Module

The harmonic rhythm of Robinflock is similar to that of Robin: each chord corresponds to a bar, and every eight chords/bars the system is forced to cadence to a tonic, dominant or subdominant (I, IV, V). The introduction of cadences allows to coherent music harmony, while maintaining real time flexibility.

3.2 Rhythm Generation Module

The Rhythm Generation Module (RGM) manages the rhythm of each musical line autonomously, thereby it is composed of four different instances of the same agent.

During the design process, speed was identified as one of the musical parameter that should be manipulated autonomously on each voice. To keep coherence among the four voices, we decided to express speed as note density. Changing the value of the BPM on each voice would indeed have resulted in unwanted results. The rhythm density ranges from a continuum of sixteenths notes up to a four-quarters note, with a simple metric division (no tuples are adopted).

To reduce the latency, thus guaranteeing a real time response, the rhythm is computed at each quarter (four times each bar). Nevertheless, for low note density situations, it could be necessary to have two, three, and four quarters notes. Consequently, the length of the note has to be modified while the note is playing. The length of the note is thereby not computed at the beginning of the notes but defined at the end, allowing continuous access to the note density parameter.

To this end, the rhythm was encoded using note ON-OFF messages. Given that each line is rhythmically computed autonomously we use the note ON message also to terminate the previous note. The ON-OFF messages are stored in an array - the Rhythm Array - of four Boolean values. Each position of the array corresponds to a sixteenth notes: 1 means new note; and 0 means that the previous note continues to play. For example, the array [1,1,1,1] corresponds to four sixteenths note and the array [1,0,0,0] corresponds to one quarter note (Figure 2 and Figure 3).

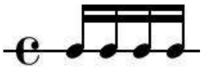


Figure 2. Rhythmic pattern corresponding to the array [1,1,1,1].

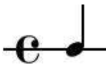


Figure 3. Rhythmic pattern corresponding to the array [1,0,0,0].

The density can range between an entire 4/4 bar of sixteenths, and a minimum of four quarter notes. The corresponding array for one entire bar are [1,1,1,1] [1,1,1,1] [1,1,1,1] [1,1,1,1] for the sixteenths (maximum density), and [1,0,0,0] [0,0,0,0] [0,0,0,0] [0,0,0,0] for the four quarter note (minimum density).

The continuum that ranges between the two extremes is discretised in sets of possible rhythms. Each set refers to a basic note length: two quarter note, one quarter note, eighth note and sixteens note. The sets contain regular and irregular patterns, for instance the sixteenths note, contains [1,0,1,1] but also [1,1,0,1] and quarter notes set contains [1,0,0,0] but also [0,0,1,0]. According to the selected density a pattern is chosen from the corresponding set.

To avoid illegal syncope, the RGM saves information on the previous quarter and which quarter of the bar is currently being computed. For instance, if the density value suddenly jumps from maximum to minimum the system would compute the succession [1,1,1,1] [0,0,0,0] that produces illegal rhythm. In this case the system bypasses the rhythm density and computes: [1,1,1,1] [1,0,0,0].

3.3 Melodies generation module (MGM)

At each quarter, the Melody Generation Module (MGM) receives the four Rhythm Arrays and the current chord from RGMs and HGM, and computes the four melodies arrays, one for each voice. The MGM outputs four streams of MIDI notes.

The computation of the notes of the arrays occurs in two steps.

1. The downbeat notes of the four voices (first position in the position in the Rhythm Array) are filled with notes of the chord, as in first species counterpoint. The first voice to be computed is the bass: it can be the fundamental or the third note of the chord. This note cannot be the fifth of the chord to avoid the second disposition of the chord, which was not used in polyphonic music [19]. Then, the other voices are computed in the following order: tenor, alto, and soprano. A chord note is assigned to each voice. If the current Rhythm Array corresponds to the first quarter of the measure, and consequently to a new chord, the algorithm checks that it does not produce a parallel octave or a fifth interval with the previous chord in a two-steps recursive procedure. In the first step, the system selects a note of the chord; in the second step, the system checked that this note does not produce octave or fifth with the previous quarter. If the selected note produces illegal parallel octave or fifth, the algorithm goes back to step one, choosing a different note. For the second, third, and fourth quarter of the measure, the chord is the same, and the melodies reposition chord notes. As a consequence, it is not necessary to pay attention to the illegal octaves [31]. Having the harmony stable for four quarters was not common in historic counterpoint. However, we adopted this solution to guarantee the flexibility of the note length while keeping the harmony coherent.
2. The other notes of the four voices are computed. The algorithm enters the second step only if the density of the voice is higher than a quarter. Two different main cases are identified: i) *syncope*: as the chord is the same, the algorithm computes the note as a consonant arpeggio; ii) *non-syncope*: the second eight is computed with a chord note, and the sixteenth upbeat notes are computed as passage notes.

Given the real-time constraints, no prediction on successive notes can be made. Thus, in the current version of the system, some elegant ornamentations of the fifth species counterpoint (e.g. dissonant suspensions, neighbour note, and double passing tones) are not implemented.

3.4 Synthesiser

The synthesiser is realized using simple FM synthesis with a different carrier modular ratio for the nearby voices, (bass FM ratio 2, tenor FM ratio 3, alto FM ratio 2, soprano FM ratio 3). The choice of using the FM and avoiding samples were suggested by the expert in music pedagogy during the design process. Using a real instrument timber could indeed cause the child to be positively or negatively emotionally influenced by a specific instrument, and this could decrease his attention over the specific exercise (e.g. the sound of a piano for the daughter of a pianist).

The synthesiser controls both the volume and the articulation. The articulation is manipulated by changing the ADSR (attack, decay, sustain, release) parameters, in the range that continues between *02 02 0 0* for the staccato and *200 100 1 100* for the legato. These values follow the standard ADSR (attack, decay, sustain, release) codification of the Max-MSP object. Attack, decay, and release are expressed in milliseconds. The sustain is a multiplicative factor used to calculate the volume during the sustain of a note, its value is normalized between 0 and 1.

4. ROBINFLOCK IN THE CONTEXT OF CHILD ORCHESTRA

This section shows a field application of Robinflock. We present the practical adoption of Robinflock in a kindergarten and report the main observations collected in this context. Full details on the field study that involved Robinflock can be found in a dedicated paper [9].

4.1 Activities

Robinflock was used over a period of seven months in a local kindergarten. A total of 27 children (14 females) aged 3 and 4 took part in the study. They were divided into two groups: 15 children of 3 years old (4 males) and 12 children of 4 years old (9 males). Both groups followed the same cycle of 12 lessons, which were facilitated by a music teacher with a specific expertise in music pedagogy for pre-schoolers. One kindergarten teacher was also present at each lesson to supervise the activities.

For this field study two different use of the system have been adopted. In the first half of the lessons, Robinflock was controlled by the music teacher via a mobile application. In the second half of the lessons, the system was directly operated by the children by means of their movements, which were tracked via motion tracking (details on the interfaces below). The observations were conducted by three of the authors.

4.2 Interfaces

Two different interfaces were developed to control Robinflock: a web app, which allows the teacher to remotely manipulate the music using a smartphone, and a motion tracker, which allows the children to directly manipulate the music.

4.2.1 Web app

The web app was used by the teacher in the first phase of the field study to help children familiarise with the music played by Robinflock and to propose a number of exercises that were specifically centred on polyphony.

The implementation of the app was based on web application technologies (node.js and javascript), which allow compatibility with smartphones and tablets. The capability of interacting wirelessly through a smartphone allowed the teacher to freely move in the room and to offer support to children. The application was implemented using sliders and buttons typical of mobile UI.

4.2.2 Motion tracking

Following the suggestions of the related studies that stressed the importance of movement to stimulate music learning [14, 18, 36], we implemented an interface that involved body movements to control the music. We adopted a solution based on accelerometers and gyroscope, which were embedded on a small (3x4 cm) and lightweight Bluetooth board (Texas instrument CC2650) powered by a small coin battery. The sensors were hidden inside a number of colourful cardboard objects aimed at attracting children's attention such as a crown and a magical wand.

The mapping between the movement of the children and the music was discussed with the experts and adapted according to the observation of the system. Initially, we planned to map the speed of melodies with the speed of the children, and the articulation with the distance of children's barycentre to the floor. During the field work, we realised that the manipulation of more than one parameter would have been too demanding for the children. As a consequence, we maintained only speed.

4.3 Observation

The experience in the kindergarten was evaluated integrating observations, interviews, and more than 300 drawings sketched by children at the end of each session. We summarise here the main considerations. Full detailed on the methodology adopted for the analysis can be found in [9].

The first important observation is that children identified and reacted properly to the three parameters. We observed how children reacted to changes of musical parameters using specific exercises designed along with the music teacher. We asked children to change their behaviour, for instance jump outside or inside a carpet, following the changes of the music. Volume, was the most difficult parameter for some of the children, as they struggled performing the exercise. By contrast, differences in articulation and speed were immediately perceived.

Robinflock was also particularly successful in helping children to familiarise with polyphonic music. This fact was assessed by observing how children reacted to the changes of specific musical lines. For the polyphony we also asked children to change their behaviour following the changes of the music, in this case different child were asked to follow the changes of different voices.

Combining the observation of children behaviour with the discussion with the music teacher allowed us to understand children that children had difficulties to manipulate more than one parameter. We also observed that a four lines polyphony was too complex to be manipulated by children. For this reason, we reduced the polyphony to two voices.

Despite these limitations, the observations revealed that Robinflock succeeded in encouraging children to actively participate in music making. Comments collected from the children confirmed this finding, as they commented on the experience stating that "I did the music" and "we played the music".

To summarize, Robinflock proved helpful to familiarise children with polyphonic musical structures and peculiarities.

5. DISCUSSION

In this paper. we presented an algorithmic system that generates polyphonic music in real time developed for interactive scenarios with children. With respect to other attempts to algorithmically compose polyphonic music, we introduced the possibility to manipulate in real time the generated music. Due to the real-time necessity, we needed to come to terms finding a trade-off between accuracy of the counterpoint and the responsiveness of the system. With respect to other systems, the accuracy is indeed reduced because of the evolution of music cannot be anticipated. Despite of this, this interactive system offered several benefits.

We propose that algorithmic composers can be adopted to help children to familiarise with polyphonic music: delegating part of the complexity of composing and performing polyphonic music to a computer allowed children to experience the control of this complex musical structures. In particular, Robinflock proved useful in helping children to recognise different lines in polyphonic music.

Reflecting on our experience with Robinflock, we propose that the development of algorithmic system for interactive scenarios should be based on the specific needs of the specific scenario, considering for instance, the target users, the environment. These needs were necessary to define the characteristic of the music and the parameters to be manipulated. For instance, Robinflock was influenced by the necessity of having a variety of music stimuli and by the necessity of manipulating three musical parameters (volume, speed, and articulation) independently for each voice.

The necessity of offering a variety of music stimuli is reflected in the HGM, which computes music in all the major and minor keys and in all the Gregorian to widen the music variety.

With respect to the three parameters, we shown that volume and articulation can be managed by a synthesiser, whereas speed has to be manipulated by an algorithmic composition system.

The necessity of manipulating the speed in real time influenced the decision to express speed as the density of the notes in the Rhythm Array. This choice allows different voices to have different behaviours. For instance, bass and soprano voices can be very slow, performing four quarter length notes, while the alto and tenor are performing sixteens notes. The manipulation of speed is probably the most significant contribution of this algorithmic system.

The interactive necessity introduces the need to determine the length of the note while the note is performing as opposed to computing it at the beginning of the note. This specific requirement leaded us to develop the ON-OFF Boolean note encoding that influenced all the RGM structure, and, to the best of our knowledge is a novelty in algorithmic composition of polyphonic music.

6. FUTURE WORKS

A number of solutions described in this paper are grounded on the specific needs of our target users. Future implementations of the system could aim at targeting different ages, for instance an adult population. This study would test the system in different interactive contexts and study how to adapt musical features and the interactive parameters to fulfil different needs.

From a musical point of view, the main limitation of the current implementation of Robinflock is the absence of elegant ornamentations of the fifth species counterpoint. Future version of the algorithm could include more complete counterpoint rules and include polished ornamentations such as dissonant suspensions, neighbour note, and double passing tones.

Acknowledgments

This paper was supported by Inf@nzia DIGI.Tales 3.6, an Italian Ministry of Education project aiming to renewing and updating preschool (3-6 years old) education programmes with ICT. We are grateful to the artworks designer Adriano Siesser, and the music teacher MariaPia Molinari for her guidance throughout the study as well as to the teachers and the children of the Kindergarden in the school of Cavedine. This research was also partially supported by EPSRC under grant EP/N005112/1.

7. REFERENCES

[1] A. R. Addessi, and F. Pachet, "Experiments with a musical machine: musical style replication in 3 to 5 year old children." in J. British Journal of Music Education 22.01, 2005, pp. 21-46.

[2] G. Aguilera, J. L. Galán, R. Madrid, A. M. Martínez, Y. Padilla, and P. Rodríguez, "Automated generation of contrapuntal musical compositions using probabilistic logic in derive." in J. Mathematics and Computers in Simulation 80.6, 2010, pp. 1200-1211.

[3] C. Aspromallis, and N. E. Gold, "Form-Aware, Real-Time Adaptive Music Generation for Interactive Experiences Data." in Proc. Int. Conf. Sound and music computing (SMC2016), Hamburg, 2016, pp 33-40.

[4] A. N. Antle, M. Droumeva, and G. Corness, "Playing with the sound maker: do embodied metaphors help children learn?." in Proc. Int. Conf. Interaction design and children, ACM, 2008, pp. 178-185.

[5] J. D. Bolter, and D. Gromala, "Transparency and Reflectivity: Digital Art and the Aesthetics of Interface Design." in J. Aesthetic computing, 2006, pp 369.

[6] W. Chai, B. Vercoe, “Folk music Classification using Hidden Markov Models” in Proc. Int. Conf. Proc. of ICAI 01, 2011.

[7] N. M. Collins, Towards autonomous agents for live computer music: Realtime machine listening and interactive music systems. Diss. University of Cambridge, 2006.

[8] D. Cope, Virtual music: computer synthesis of musical style. MIT press, 2004.

[9] C. Core, A. Conci, A. De Angeli, R. Masu, F. Morreale, "Designing a Musical Playground in the Kindergarten" to appear in Proc. Int. Conf. British HCI, 2017.

[10] P. Donnelly, and J. Sheppard, "Evolving four-part harmony using genetic algorithms." in Proc. Int. Conf. European Conference on the Applications of Evolutionary Computation, Springer Berlin Heidelberg, 2011, pp. 273-282.

[11] J. D. Fernández, and F. Vico, "AI methods in algorithmic composition: A comprehensive survey." in J. Journal of Artificial Intelligence Research 48, 2013, pp 513-582.

[12] Gartland-Jones, Andrew, "MusicBlox: A real-time algorithmic composition system incorporating a distributed interactive genetic algorithm." In Workshops on Applications of Evolutionary Computation, Springer Berlin Heidelberg, 2003, pp. 490-501.

[13] J. Glover, and S. Young, Music in the early years. Routledge, 2002.

[14] E. Gordon, A music learning theory for newborn and young children. Gia Publications, 2003.

[15] D. Herremans, and K. Sörensen, "Composing first species counterpoint with a variable neighbourhood search algorithm." in J. Journal of Mathematics and the Arts 6.4, 2012, pp. 169-189.

[16] L. Hetland, "Listening to music enhances spatial-temporal reasoning: Evidence for the" Mozart Effect"." in J. Journal of Aesthetic Education 34.3/4, 2000, pp. 105-148.

[17] Jr. Hiller, A.Lejaren, and L. M. Isaacson, "Musical composition with a high speed digital computer." Audio Engineering Society Convention 9, Audio Engineering Society, 1957.

[18] E. Jaques-Dalcroze, Rhythm, music and education. Read Books Ltd, 2014.

[19] K. Jeppesen, Counterpoint: the polyphonic vocal style of the sixteenth century. Courier Corporation, 2013.

[20] S. Kalénine, and F. Bonthoux, "Object manipulability affects children’s and adults’ conceptual processing." in J. Psychonomic bulletin & review 15.3, 2008, pp. 667-672.

[21] G. E. Lewis, "Interacting with latter-day musical automata." in J. Contemporary Music Review 18.3, 1999, pp. 99-112.

[22] A. Mann, The study of fugue. Courier Corporation, 1987.

[23] V. P. Martín-Caro, and D. Conklin, "Statistical Generation of two-voices florid counterpoint” in Proc. Int. Conf. Sound and music computing 2016, pp. 380-387.

[24] F. Morreale, A. De Angeli. Collaborating with an Autonomous Agent to Generate Affective Music in J. Computers in Entertainment, ACM, 2016.

[25] F. Morreale, A. De Angeli, R. Masu, P. Rota, and N. Conci, “Collaborative creativity: The music room.” In J. Personal and Ubiquitous Computing, 18(5), Springer, 2014, pp.1187-1199.

[26] F. Morreale, R. Masu, and A. De Angeli. "Robin: an algorithmic composer for interactive scenarios Proc. Int. Conf. Sound and music computing (SMC2013) Stockholm, 2013 pp. 207-212.

[27] F. Morreale, R. Masu, A. De Angeli, and P. Rota, “The music room” in Proc. Int. Conf. CHI'13 Extended Abstracts on Human Factors in Computing Systems. pp. 3099-3102, ACM.

[28] G. Nierhaus, Algorithmic Composition: Paradigms of Automated Music Generation. Springer, 2009.

[29] F. Pachet, "The continuator: Musical interaction with style." in J. Journal of New Music Research 32.3, 2003, pp. 333-341.

[30] S. Phon-Amnuaisuk, and G. Wiggins, "The four-part harmonisation problem: a comparison between genetic algorithms and a rule-based system." in Proc. Int. Conf. AISB’99 Symposium on Musical Creativity, London: AISB, 1999, pp. 28-34.

[31] W. Piston, Harmony. Norton, 1948.

[32] J. J. Polito, M. Daida, and T. F. Bersano-Begey, "Musica ex machina: Composing 16th-century counterpoint with genetic programming and symbiosis." in Proc. Int. Conf. International Conference on Evolutionary Programming, Springer Berlin Heidelberg, 1997, pp. 113-123.

[33] D. Pond, "A composer's study of young children's innate musicality." in J. Bulletin of the Council for Research in Music Education,1981, pp. 1-12.

[34] R. Rowe, "Machine listening and composing with cypher." in J. Computer Music Journal 16.1, 1992, pp. 43-63.

[35] A. Schoenberg, Theory of harmony, Univ. of California Press, 1978.

[36] W. L. Sims, "The effect of high versus low teacher affect and passive versus active student activity during music listening on preschool children's attention, piece preference, time spent listening, and piece recognition." in J. Journal of Research in Music Education 34.3,1986, pp. 173-191.

[37] J. M. Thresher, "The contributions of Carl Orff to elementary music education." in J. Music Educators Journal 50.3, 1964, pp. 43-48.

[38] R. Jackendoff, A generative theory of tonal music. MIT press, 1985.

[39] I. Xenakis, Formalized music: thought and mathematics in composition, Pendragon Press, 1992.

[40] K. Wassermann, M. Blanchard, U. Bernardet, J. Manzolli, and P. F. Verschure. “Roboser-An Autonomous Interactive Musical Composition System.” Proc. Int. Conf. ICMC, Berlin, 2000.

Technology-Assisted Performance of Polytemporal Music

Philippe Kocher
Institute for Computer Music and Sound Technology
Zurich University of the Arts
philippe.kocher@zhdk.ch

ABSTRACT

This paper is an intermediate report of an ongoing artistic research project on technology-assisted performance practice. It describes the application *Polytempo Network* and some works that were realised using this application in the course of the last two years. The different compositional approaches chosen for every individual work and the experiences gathered during the process of composing, rehearsing and performing these works are discussed. The leading question is to what extent the usage of this technology is aesthetically significant.

1. INTRODUCTION

The application *Polytempo Network* has been developed at the ICST Institute for Computer Music and Sound Technology in Zurich [1]. It facilitates the synchronisation of musicians in polytemporal relations and/or over larger distances that would not allow for direct interaction or visual communication. For this purpose multiple instances of this application run on computers linked in a local area network.

The application displays the score and indicates the tempo by two animated bars at the top and left edge of the screen. The movements of these bars resemble remotely the gestures of a conductor: a downward movement indicates a downbeat, a sideward movement any other beat (Figures 1 and 2). For most musicians a tempo visualisation has several advantages compared to a traditional acoustic click track, for instance, it does not interfere with the musicians hearing sense and the animation allows for a better anticipation of beats and tempo changes. However, if a musician prefers the acoustic click track, the possibility to output an acoustic metronome or to send the beats as Midi note messages is provided as well. Moreover, as the application does not only indicate the tempo on the screen but also displays the score, it lends itself equally well to those aleatoric compositional concepts known as 'open' or 'mobile' form.

The motivation behind the development of this application is the research question, whether the availability of a versatile technology to synchronise performers can possibly lead to novel forms of compositional strategies. Or, expressed as a question put towards composers: how to conceive of a music that exploits the specific potential of such a technology?

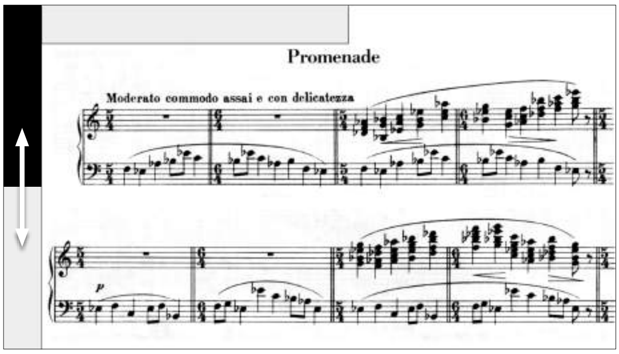


Figure 1. A downbeat is indicated by a bar moving down and up again at the left edge of the screen. This vertical movement is similar to a conductors downbeat.

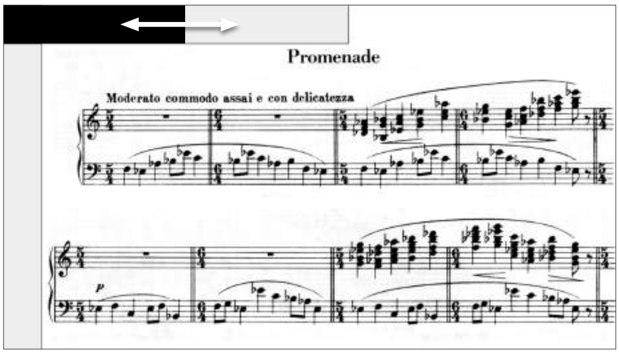


Figure 2. Any other beat is indicated by a horizontal movement of the bar at the top edge of the screen.

2. CONTEXT

2.1 Music History

The first use of technology to synchronise musicians in a performance situation goes back to the middle of the 19th century. For the second edition of his treatise on instrumentation, published in 1855, Hector Berlioz added a chapter on conducting entitled *le chef d'orchestre* [2]. In this chapter he addresses the issue of keeping instrumentalists or singers, which are placed behind the scenes in an opera, in time. He describes an electrical device, the *métronome electrique*, to transmit the conductor's beat to off-stage musicians. Berlioz came across this machine at a theatre in Brussels and used it again for a monster concert at the Paris Universal Exhibition in 1855 to transmit his beat to five sub-conductors. As this technology-assisted conducting was obviously not compliant with the 19th century notion of musical romanticism, this device disappeared again. In the early 20th century, we find several attempts to synchronise music and imagery in the cinema, for which several visual and acoustical solutions to indicate the tempo were invented [3].

As far as tempo polyphony is concerned, there are several works in the 20th century classical music that abandoned the notion of a common tempo for all performers, of which the earliest examples can be found in the oeuvre of Charles Ives. In these works different tempos are assigned to individual instruments or musical layers and in most cases the tempos are precisely indicated by metronome marks. Yet the actual coincidence of musical events remains speculative, since no human is able to hold a given tempo accurately. Hence, these pieces explore primarily the expressive quality of musical stratification *as such,* and, of course, the composers were aware of this and took the appropriate compositional measures.

The possibility to keep all performers in sync even in intricate polytemporal relations, and even when the tempos accelerate or decelerate independently, opens the door for interesting polyphonic strategies. However, it requires a technology-assisted performance practice, i. e. the usage of technical means to keep the tempo to allow for an appropriate timing accuracy. Metronomes or click tracks are commonly used in such situations. The first description of a polytemporal click track was given by Emmanuel Ghent in 1967 [4, 5].

2.2 Related Projects

Networked music performance has been considered since a few decades in contemporary music composition. And it has developed rapidly due to the advance of the enabling technologies [6]. A couple of existing systems provide networked and interactive display of music notation: *Quintet.net* enables an ensemble to share notation and other information across a network of laptops, whether the musicians are in the same room or spread across the world [7]. It is built on top of MaxMSP, and the notation is rendered with *MaxScore* [8]. *INScore* can display images, vector graphics, and symbolic music notation, for which the Guido Music Notation or MusicXML formats are used [9]. It indicates the current position in the (graphical or symbolic) score by a cursor that advances with the appropriate speed. The *Decibel ScorePlayer* is an iOS application that facilitates the presentation and network-synchronised scrolling of graphical scores on multiple iPads [10].

The mentioned projects allow for different forms of non-traditional scores as well as scores generated in real time. However, even though they might support polytemporal music to a certain degree, they are not specifically designed for it. The application *Polytempo Network* fills this gap.

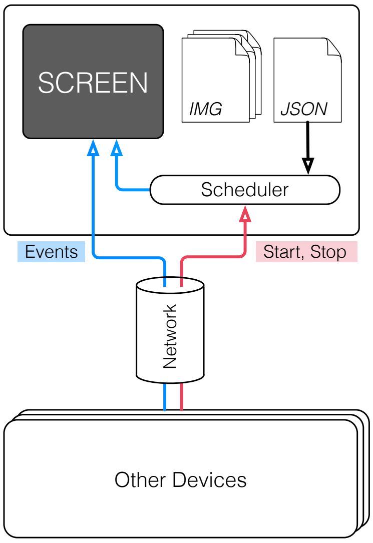


Figure 3. Simplified technical diagram. Events are either read from the JSON-formatted ‘electronic score’ and processed by the built-in scheduler, or received as OSC messages over the network.

3. TECHNICAL DETAILS

Polytempo Network is a standalone application for Mac OS and Windows available for download from the project’s website. [1] Dependent on the tempo structure of the music, and especially on the fact whether the form is predefined or generated in real time, the application can be operated in two different modes: stand-alone and message-driven. The former mode employs a built-in scheduler to play though a given score, and is typically used for compositions with a predetermined tempo and a linear time structure in a traditional fashion. The latter mode requires a central computer to control the flow of time for all instances and is used for music whose tempo and/or form is subject to changes in real time.

3.1 Standalone Operation

For the performance of a composition with a linear, yet possibly polytemporal, time structure the typical procedure is as follows: *Polytempo Network* has to be provided with two types of data, the sheet music stored as image files and an ‘electronic score’ in JSON format (Fig. 3). The electronic score contains a timestamped lists of events. This list is processed by an built-in scheduler, which executes all the events when they are due. By sending commands over the network to start or stop the scheduler several instances of *Polytempo Network* are kept in sync. An useful feature in rehearsal situations is the possibility to jump to and start from an arbitrary point within the piece. It does not matter if the musicians have to play in different tempos: to resume the music at any given point in time, each instance starts its conducting at the next possible downbeat and, if necessary, with the appropriate time-delay.

Among the events listed in the electronic score the most important are obviously those that represent beats and

[1] polytempo.zhdk.ch

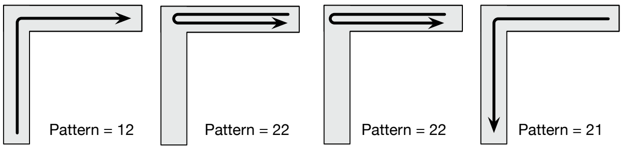


Figure 4. The *pattern* of a beat is a two-digit number that indicates the initial and final position of the bar and hence the direction of movement of the animation. The four beats of a 4/4 bar would have the patterns 12, 22, 22, and 21.

those that indicate that a certain section of an image file has to be shown on the screen. The latter type of event is used to display the music and to automatise the page turning. The former type is defined by the parameters *duration* and *pattern*, both of which influence the rendering of the on-screen animation. The pattern is a two-digit number where the first digit specifies the initial position and the second digit the final position of the animation. The values of these digits are either 1 = bottom left or 2 = top right (Fig. 4).

A JSON object that represents a beat event would, for instance, be written as follows:

```
{"beat":{"time":13.5,"duration": 0.6,
 "pattern":12}}
```

This defines a downbeat (as the pattern 12 indicates) with a duration of 0.6 s and scheduled to be executed 13.5 s after the beginning of the piece.

3.2 Message-driven Operation

In the message-driven operation mode the built-in scheduler remains inactive and all functionalities are controlled by messages received over the network. The communication protocol is Open Sound Control (OSC). Every event that can appear in the electronic score has an equivalent in form of an OSC command.

For this purpose the basic OSC format is adapted in the following way: The address part of the OSC message simply indicates the event's name and the parameters consist of key-value pairs. The same beat event as above would look as follows:

```
/beat duration 0.6 pattern 12
```

As there is no indication, the event's time is 'now' and the animation to indicates a downbeat would be executed on-screen immediately after this message has been received. Thus, the flow of the music can entirely be controlled in real time from any external OSC capable application (e. g. SuperCollider or MaxMSP). This allows for compositional strategies that employ non-linear, real-time generated or interactive forms.

4. WORKS

This section describes five works by the author and other composers that were realised during the last two years. This list is not exhaustive, it only covers those works that made the most substantial use of the application *Polytempo Network*.

Figure 5. Performance of *Trails 1*. Three musicians are synchronised to an audiovisual playback.

4.1 Trails I (2011)

The composition *Trails I* by the author in collaboration with Daniel Bisig exemplifies a straightforward usage of the application *Polytempo Network*: the synchronisation of performers to an audiovisual playback (Fig. 5). Of course, this does not draw on the full potential of the application, as there is no tempo polyphony involved whatsoever. Still, as this was one of the earliest pieces realised with this application, it gave the opportunity to address some fundamental questions about its practicality. For the premiere of this piece in 2011 a prototype programmed in MaxMSP was used. It was only for a restaging in the fall of 2015 when this prototype could be exchanged with the standalone application.

The realisation of this piece gave rise to the question whether such an automated conducting feels awkward to the musicians. In this kind of technology-assisted performance practice, the musicians are required to follow the beat as it is given by a machine, which possibly renders the music lifeless and mechanical. But these worries were proved wrong: until now, *Trails I* has been performed by four different ensembles and none of the musicians ever complained. In general, musicians specialised in contemporary music are familiar with the fact that a conductor does not convey emotions but rather 'organise' the music by soberly beating the tempo. Therefore, they did not feel hindered by the electronic conductor. On the contrary, the performers expressed their appreciation for the beat being so clear and unambiguous, which they found helpful for the performance of rhythmically complex music.

4.2 Quadrat (2015)

The piece *Quadrat* for two trumpets and two trombones was written by the author with the intent to explore the application's facility to enable tempo polyphony and independent tempo progressions. Furthermore, in order to emphasise the independency of the parts, the brass players were positioned at the four corners of the auditorium, that

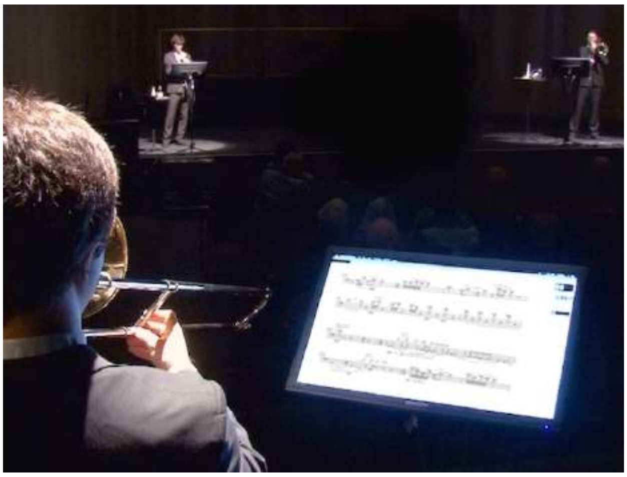


Figure 6. Performance of *Quadrat*. The musicians are positioned at the four corners of the auditorium, approximately 20 m apart.

is, approximately 20 metres apart from each other (Fig. 6).

Especially the first section of this piece exhibits a constant change back and forth between synchronicity and independence. Every few bars, all instruments play a note exactly together. Between these notes they execute rhythmical structures in different tempos or in tempos slowly drifting apart. This structure was well perceivable by the listener as a (tempo-)polyphony interspersed with synchronous chords.

Concerning the performance practice, the musicians had to develop a special (but quickly mastered) skill. Being positioned wide apart, they had to precisely adjust their timing solely to the beats indicated by the application, rather than using their hearing sense to adapt their playing to the other musicians. However, other aspects of musical interaction like control of intonation or dynamic balance remained the same as in traditional musical practice.F

One musical texture posed a problem: At some points, the four musicians have to play interlocking rhythms, which should result in a rhythmic pattern of regular semiquavers circling around the audience. In practice, this was not satisfactorily realisable for two reasons: On the one hand, playing a tight interlocking rhythm does not simply need the aid of a metronome but rather a very subtle interaction between the performers, which cannot be established over large distances. On the other hand, the visual animation to indicate the tempo is less precise than the punctual ‘beep’ of a click track. In most other cases this is a considerable advantage, as it does not give the musicians the feeling that they have to fight against the metronome when they get a litte behind or ahead of the beat. But for the execution of tight rhythms this flexibility is not particularly helpful.

4.3 Manduria (2016)

Without knowledge of the aforementioned piece, the composer Stefan Wirth came up with a similar idea of spatial distribution of the musicians for his piece *Manduria*. Seven groups of one to three musicians each were placed around the audience (Fig. 7). The immersive sound thus produced was very impressive, not only because of its spatial quality but also due to the fact that a dense polyphonic structure or a superimposition of differently characterised musical layers can be segregated more easily if the sound sources are distributed in space.

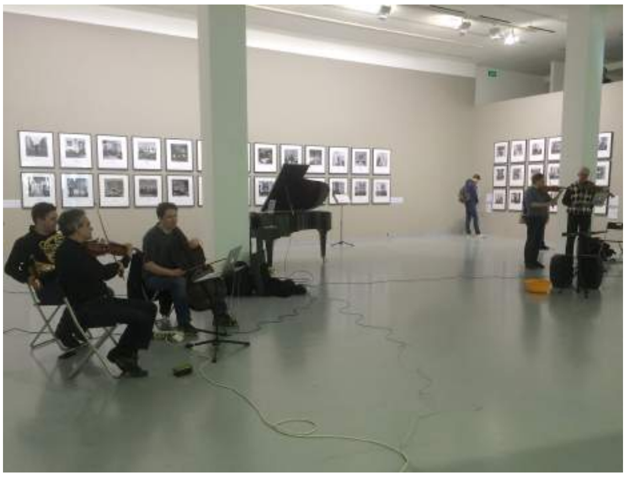


Figure 7. Performance of *Manduria*. The musicians are grouped in small sub-ensembles and distributed in a gallery space.

In accordance with this, it was Wirth’s intention to assign an individual characteristic to every instrumental group and to produce a very heterogeneous texture. This lead to the question if in fact spatialisation is musically more effective than tempo polyphony and therefore will turn out to be the primary purpose of the application *Polytempo Network* in the future. Although it is too early to make a final assessment, one can certainly say that spatial effects are more straightforwardly composed than tempo polyphonies, which always involve a great deal of tedious mathematics.

Wirth, being an accomplished composer, relied on his experience and wrote a piece that worked well and was successfully performed, but he did not engage in experiments with complex tempo structures. The tempo polyphony in *Manduria* consists of the superimposition of speeds that can be expressed in notation, e. g. a 4/4 bar in the same time as a 5/4 bar, etc. Furthermore, the basic tempo MM = 72 is present throughout the piece and all other tempos are proportionally related to it. This allowed Wirth to plot his composition in a traditional score with occasionally coinciding bar lines. That way, he could stay in his familiar working environment and maintain his habitual workflow, which eventually was necessary to deliver a piece of high quality and in due time.

4.4 Tempo Studie (2016)

The short composition *Tempo Studie* was the author’s attempt to realise a piece with an open form generated in real time. This trio for violin, bass clarinet and accordion utilises a computer algorithm that randomly chooses fragments and presents them to the performers. But not only the succession of fragments is random-based but also the

Figure 8. Rehearsal session of *Tempo Studie*. The musicians play individual tempo progressions that are random generated in real time.

tempo, which is individually calculated for every part. In addition, the tempos never remain static but constantly accelerate and decelerate, which means that the three musicians have to play in different and independently changing tempos throughout the piece. Yet, in order to establish appreciable relations between the parts, prominent features in the musical texture, e. g. a sudden changes in loudness, density, or pitch range are synchronised ever so often. To enable these coincidences, the random algorithm calculates the tempo progressions accordingly [11].

The realisation of this piece omitted *Polytempo Network's* built-in scheduler and made use of the message-driven operation mode. All events were remotely controlled via OSC: the individual beats to indicate the ever-changing tempos and the display of graphic files i. e. the music of the currently chosen random fragment. All computers were connected in a network and controlled from an additional fourth computer. The calculations to generate the arrangement of fragments and the appropriate tempo progressions were implemented in the programming environment SuperCollider. The overall duration of one complete run-through of this piece is only about two and a half minutes, which allows for several renditions in a row and for an immediate comparison of the different versions.

The musicians were challenged in two ways when performing this piece: first, they had to be able to play the same music in different tempos (the fastest possible tempo being more than twice as fast than the slowest) and adapt to the different gestural qualities thus revealed. Second, they had react quickly and spontaneously to the ever changing tempo indicated on the computer screen. The musicians did not only accept this challenge, but also found it rather inspiring, as it appealed to their expertise as interpreters as well as improvisors. For the listeners, the polyphony of tempos lead to the impression that the performers were improvising along with each other, whereas the perceivable coincidences added a clear structure. This combination of both qualities, freedom and structure, turned out to be musically quite effective.

4.5 The Same (Not) The Same (2017)

The most recent work is the piece *The Same (Not) The Same* for eight instruments by André Meier. This composition takes advantage of technology in several ways: not only to distribute the musicians in space – strings, piano and percussion on stage, woodwinds around the audience – but also to execute several random choices in real time. Some sections of the piece exist in multiple variants and the one to be performed is chosen by a random algorithm. At some points a continuous layer of music is occasionally overlapped with shorter fragments chosen by a random algorithm. Several sections of the piece are to be played in a random tempo chosen from a given range of tempos. The beginning of every overlapping fragment, also played in a random tempo, is placed in such a way that the music of the fragment and the continuous layer coincide at a certain point. To realise these compositional concepts, *Polytempo Network* is used in its message-driven operation mode, controlled from a central computer.

How such a randomisation could be made appreciable for the listener rather than remaining a speculative concept, was one of Meier's main considerations. Hence, one section of the piece consists of one fragment repeated over and over again. As the iterations of this fragment are not completely identical but vary in tempo and some other details, the listener is given the chance to get a vague notion of the pieces underlying concept.

5. CONCLUSION AND OUTLOOK

The experience gained from the performances of the pieces described in this paper was essential, as it provided knowledge about the efficiency of several conceptual ideas when realised in practice. This knowledge is particularly valuable for composers who intend to use *Polytempo Network* for their music in the future.

The case of *Manduria* showed the dialectic relationship between notation and composition. The decision to remain within the conventions of traditional notation provided security but hindered the exploration of new concepts. In order to encourage composers to delve into experimental tempo structures, it seem important to provide the appropriate tools, especially to alleviate the burden of mathematical calculations with which the realisation of complex tempo structures is always associated [12].

There are three aspects where the use of technology is significant, that is, enables aesthetic results that could not be achieved in a performance practice that is not technology-assisted. First, the distribution of musicians in space while keeping them rhythmically synchronised. In fact, spatiality was included in several works, which indicates that composers find it attractive. Second, the simultaneity of musical layers in different tempos, or even different tempo progressions, interspersed with coincidences of salient musical events is effective, as it can be appreciated as a balance between rhythmical freedom or independency and synchronisation. Third, the possibility to generate and perform 'open' form compositions in real time. It is, however, necessary that the whole piece or certain parts of it are played several times in a row, in order to enable the comparison of the different variants and consequently the appreciation of the random-based form.

The question whether the availability of technology leads to novel forms of compositional strategies, posed at the beginning of this paper, has to be further explored. This requires more composers with different backgrounds to engage with this technology and create works that can serve as application scenarios. It is of particular importance that future works will also involve different genres and styles.

As this is an ongoing project, the application *Polytempo Network* will be further developed and extended. Two features are to be added with high priority: first, the facility to annotate the score, which is much asked for by performers, and second, the possibility for wireless network communication, which would simplify the technical setup for a rehearsal or a concert a great deal.

Acknowledgments

I would like to thank my fellow composers Stefan Wirth and André Meier for their artistic contributions to this research project.

6. REFERENCES

[1] P. Kocher, “Polytempo Network: A System for Technology-Assisted Conducting,” in *Proceedings of the International Computer Music Conference*, Athens, 2014, pp. 532–535.

[2] H. Berlioz and R. Strauss, *A Treatise on Modern Instrumentation and Orchestration*. New York: Edwin F. Kalmus, 1948.

[3] T. Plebuch, “Zeitarbeit: Das Zusammenspiel von Menschen, Maschinen und Musik in der Entwicklung von Tonfilmtechniken,” in *Spiel (mit) der Maschine: Musikalische Medienpraxis in der Frühzeit von Phonografie, Selbstspielklavier, Film und Radio*, M. Saxer, Ed. Bielefeld: Transcript, 2016, pp. 177–210.

[4] E. Ghent, “Programmed Signals to Performers. A New Compositional Resource,” *Perspectives of New Music*, vol. 6, no. 1, pp. 96–106, 1967.

[5] ——, “The Coordinome in Relation to Electronic Music,” *Electronic Music Review*, vol. 1, pp. 33–43, 1967.

[6] L. Gabrielli and S. Squartini, *Wireless Networked Music Performance*. Singapore: Springer, 2016.

[7] G. Hajdu, K. Niggemann, Á. Siska, and A. Szigetvári, “Notation in the Context of Quintet.net Projects,” *Contemporary Music Review*, vol. 29, no. 1, pp. 39–53, 2010.

[8] N. Didkovsky and G. Hajdu, “Maxscore: Music Notation in Max/MSP,” in *Proceedings of the International Computer Music Conference*, Belfast, 2008, pp. 2–5.

[9] D. Fober, Y. Orlarey, and S. Letz, “Augmented Interactive Scores for Music Creation,” in *Proceedings of Korean Electro-Acoustic Music Society’s 2014 Annual Conference*, Seoul, 2014.

[10] C. Hope, A. Wyatt, and L. Vickery, “The Decibel ScorePlayer: New Developments and Improved Functionality,” in *Proceedings of the International Computer Music Conference*, 2015, pp. 314–317.

[11] P. Kocher, “Rethinking 'Open Form',” in *Proceedings of the Generative Art Conference*. Florence: Domus Argenia Publisher, 2016, pp. 339–349.

[12] ——, “Polytempo Composer: A Tool for the Computation of Synchronisable Tempo Progressions,” in *Proceedings of the Sound and Music Computing Conference*, Hamburg, 2016, pp. 238–242.

INVESTIGATING SPATIAL MUSIC QUALIA THROUGH TISSUE CONDUCTION

Peter Lennox
University of Derby, UK
P.Lennox@derby.ac.uk

Ian McKenzie
University of derby, UK
I.McKenzie1@derby.ac.uk

ABSTRACT

A spatial array of vibro-mechanical transducers for bone-and-tissue conduction has been used to convey spatial ambisonic soundscape and spatial musical material. One hundred volunteers have undergone a five-minute listening experiences, then have described the experience in their own words, on paper, in an unstructured elicitation exercise. The responses have been aggregated to elicit common emergent descriptive themes, which were then mapped against each other to identify to what extent the experience was valuable, enjoyable and informative, and what qualia were available through this technique. There appear to some substantive differences between this way of experiencing music and spatial sound, and other modes of listening. Notably, the haptic component of the experience appears potentially informative and enjoyable. We conclude that development of similar techniques may have implications for augmented perception, particularly in respect of quality of life (QoL) in cases of conductive hearing loss.

1. INTRODUCTION

This paper describes progress in investigating the experienced properties of spatial music delivered through an apparatus featuring multiple transducer locations situated on the cranium. As the listener's ears are unoccluded, residual air-conduction hearing is unaffected. This technique is a non-invasive augmentation of existing sensory capabilities; it can be efficacious in respect of the conductive component of hearing loss, but not sensorineural components.

The goal is to characterise, and subsequently quantify dimensions of perceptual experience in relation to music listening in this manner. Is the experience meaningful, in what ways, and in what ways might it differ from other music listening modes?

The aim of providing the spatial aspect of music follows the general trend in current music production and reproduction toward enhanced spatial attributes such as image focus [1], localizability, motion, spaciousness [2], and ensemble depth [3]. There may be substantive distinctions between the desirable and feasible spatial attributes of musical experience vs. those for spatial environment listening. Nevertheless, many of the experiential attributes that may be available through multi-speaker systems would not be feasible in conventional tissue or bone conduction techniques (discussed in section 4). Hence those with some degree of bilateral or unilateral hearing impairment do not have ready access to the kinds of musical experience available to listeners with unimpaired hearing.

Whilst communications difficulties in hearing impairment are receiving increasing attention, the quality-of-life (QoL) implications of music deprivation have received less. Assistive technologies for speech comprehension do not currently adapt well to music listening. [4]

In the first stage, to avoid a reductionist approach and obviate the need for trained listeners, we adopted an unstructured elicitation methodology (for discussion, see [5], [6]**)** whereby the prototype apparatus (discussed in section 5) was demonstrated over several days in various venues. Subjects were self-selecting, were given no instructions as to what to listen for and briefly recorded their initial impressions on paper after listening for approximately 5 minutes; some volunteered to repeat the experience on subsequent days. We observed variations in volunteers' responses (discussed in section 6). We did not aggregate data on known hearing impairments.

2. SENSORY AUGMENTATION FOR AUDIO MATERIAL

Approximately 5% of the World's population, that is, 360 million people, suffer from "disabling hearing loss" [7] and the proportion of over-65s rises to about 33% [8]. 13.4% of geriatric patients have significant conductive components to their hearing loss [9].

A substantial proportion of the population are subject to 'music deprivation' and inasmuch as music listening contributes to people's sense of wellbeing or "Quality of Life" (QoL), this deprivation may have significant and long-term health and wellbeing consequences. Assistive technologies implementing sensory augmentation could ameliorate the effects of lack of ready access to music, the experiential attributes of music listening can be reinstated and tangible benefits might accrue.

We distinguish sensory augmentation from sensory substitution in that the aim is to extend perception, not to substitute. However, augmentation might itself be augmented with some elements of substitution, and so the concepts overlap. Multimodal presentations of certain classes of information might provide richer experiences. Vibrotactile stimuli can be used to enhance perceived low frequency content, emphasize transients and steering of spatial auditory perception [10]. Philosophically, we think in terms of 'information channels' rather than direct sensory equivalence.

Such multimodal interactions will be subjects of future investigative work.

3. SPATIAL MUSIC

There has, in the late 20th and early 21st centuries, been burgeoning interest in spatial, surround or 3-dimensional music. The subject can be discussed in engineering, aesthetic and perceptual terms [11][12][13]. The underlying principles are that spatial (as against "non-spatial") music might provide enhanced experience in terms of involvement and immersivity. In information-transmission terms, incorporating spatial parameters facilitates greater information throughput, allowing finer detail to be depicted and discerned.

It is acknowledged that perceptual tasks in music listening differ from those in environment listening. In the latter, requirements for timely detection of threat and reward are presumed to have exerted evolutionary influence on phylogenetic development. Notwithstanding, exaptation [14], whereby evolved mechanisms or capabilities can become co-opted for other uses, provides that our spatial abilities are available for the experiencing of music.

Stereo [15] provides for a loudspeaker-feed signal set that generates interaural differences (in amplitude and phase) at the ideal listening position that can produce the powerful illusion of a left-right discriminable stage with multiple spatially-separate musical sources, either static or moving. Additionally, spatial reverberant fields (captured or synthesized) can give some sense of ensemble depth (some sources closer than others) and spaciousness. The effect is of a proscenium arch presentation. The stereo signal set can be listened to over headphones; however the effect is generally of a soundstage distributed left-right between the ears, giving a particular "in-the-head" experience. A binaural signal set can be used (either binaurally recorded or synthesized) to promote "externalization" (for a discussion see: [16]) and in the optimal case, where the head-related transfer function (HRTF) used in the production of the signal set closely matches the HRTF of the listener, strong impressions of an externalized, three-dimensional environment can ensue. However, such close matching is rarely feasible and the usual experience falls short of the theoretical optimum.

Surround sound, where a complex signal set is fed to multiple loudspeakers surrounding the listener(s) can depict many source-locations, movements and a sense of being immersed in a whole spatial environment. However, perceptions of depth-of-field (variations in perceiver-source distance) remains limited. Systems range from fairly simple (e.g. Dolby 5.1 surround) to complex (e.g high-order ambisonics or wave field synthesis).

The spatial qualia engendered by the various approaches differ; a large and complex system may well give experiences of large environments but may be less competent in producing "intimate" ones with sources close to the listener. The converse is generally the case with small, intimate systems. Composers of spatial music are thus constrained in what qualia they can attempt to offer.

For discussion of spatial music compositional concerns, see for example [17].

In all the above cases, listeners with bilateral or unilateral hearing deficits will experience degraded spatial musical qualia, reducing immersion and impairing enjoyment of the material.

4. SPATIAL TISSUE CONDUCTION

Auditory perception elicited by means of mechanical transduction, i.e. a tuning fork pressed against the cranium, has long been known. Single vibro-tactile transducers have been in use in audiology and the hearing aid industry for decades. Until fairly recently spatial audio was not thought possible through tissue conduction, theorised interaural level differences due to interaural attenuation were not considered sufficient; studies have shown this not to be the case [18][19][20]. In all three experiments to assess lateralisation, stimuli were presented bilaterally with transducers placed in contact with either the mastoid process behind the ear or the condyle just in front of the ear; all produced similar results to that of headphones. These experiments indicate that when sound is presented through tissue conduction we still make use of the same binaural cues as for air conducted (AC) sound.

Auditory localization is dependent on the physiological and anatomical properties of the auditory system as well as behavioral factors. The textbook primary cues for auditory localization are interaural differences and spectral cues [21][22][23]. The ridges and folds in the outer ear reflect and absorb certain frequency components of a sound wave, the spectral characteristics of a sound wave will differ if approaching the ear from different directions. Due to the shape of the pinnae providing this filtering effect the elevation and position of sound sources is encoded in direction-dependent spectral cues allowing us to localize sound sources. Many literary sources agree that vertical information derives exclusively from position-depending differences in the frequency filtering properties of the external ear.

Whilst interaural differences akin to air conduction may result when sound is presented through tissue conduction, no sound is presented to the outer-ear specifically the pinnae and vertical information should be absent; some comments suggest this is not the case. This anomaly may arise out of fine differences in arrival times caused by propagation along multiple signal pathways from transducer to the basilar membrane. There is also an intriguing possibility of multimodal cueing; binaural auditory cues merging with additional information provided through the somatosensory system via haptic cues [10][24][25][26]. When using a multiple transducer array vertical information is available to the listener as well as externalisation of the perceived sound; how this is the case continues to be the subject of further investigation.

5. APPARATUS, MUSICAL MATERIALS AND LISTENING CONDITIONS

5.1 Apparatus

Psychophysical investigation of the dimensions of experience of spatial tissue-conduction listening will prove useful, but for many first-time listeners, bases for comparison may be lacking; a training period targeting specific attributes may be required. Determining what those attributes might be is the aim of the present study.

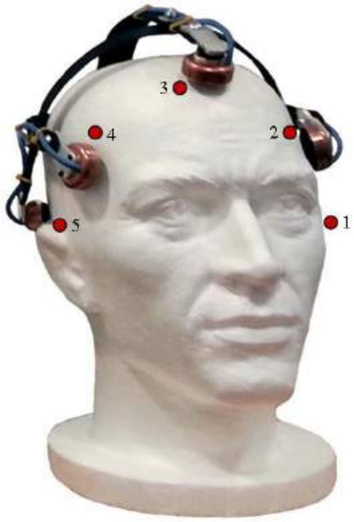


Figure 1. Multiple transducer array

Sounds presented at:

- 1-left mastoid
- 2 – 25mm above left temple
- 3: between forehead and vertex
- 4: 25mm above right temple
- 5: right mastoid

Figure 2. BCT-2 10W transducer

A prototype headset transducer array using five BCT-1 8Ω 90dB 1W/1 m tactile transducers has been used to display a range of spatial soundscapes and music. Each transducer receives a discrete signal set through an individual amplifier; a Focusrite PRO 26 i/o interface provides fire-wire connection to a mac mini running Reaper DAW. A single BCT-2 10W transducer was also available for listeners to position on the jaw, zygomatic arch or back their head/neck. A set of banded style 3M Ear Plugs were available for listeners to use and compare the experience with the plugs in vs out.

5.2 Listening Materials

Environmental and musical stimuli was processed using a variety of effects and routed in different formats; stereo, modified stereo, ambisonics and direct feed. A 1st order ambisonic recording of a country park captured using a Soundfield ™ microphone provides the ambient background; stereo recordings of bird sounds, a steam train and music alongside mono FX clips were used to create the soundscape. Signals were processed using Reaper ® DAW; signals were spatially encoded using WigWare 1st order ambisonic panning and decoded through a WigWare 1st order periphonic ambisonic decoder patched to the transducer array.

5.3 Listening Conditions

The proto-type has been on demonstration at IOA Birmingham, ICMEM Sheffield and PLASA London. At PLASA we recruited one hundred untutored listeners, with a mixture of expertise; none reported experience of tissue conduction. Auditions were of five minutes duration, no prior instructions were given and volunteers were invited to record initial reactions and observations on paper immediately after auditioning. The listening tests took place in non-ideal conditions, as part of the Exploratorium exhibit we shared the space with four other exhibitors. The Exploratorium was located on the upper level of the large exhibition hall, a large footfall and other exhibitors using amplified sound produced a considerable noise floor.

5.4 Limitations

Equipment and calibration: The transducers in use have the following known limitations:

- Frequency response: 200Hz to 16 KHz, low frequencies are not well served, resulting in a ‘thin’ sound for some musical material.
- Component matching: the manufacturers do not publish information on performance matching.

With a cohort of 100 and a wide variety of head sizes, precision in determining matched contact force for all transducers was infeasible, possibly resulting in different spatial experiences for different listeners. Additionally, as audiological testing was impossible, variations in hearing acuity could not be taken into account

The demonstrations took place in an environment with high levels of ambient sound, especially in vocal ranges, entailing concomitant constraints on dynamic range and hence subtlety of detail.

The method of recording responses proved to be suboptimal, as many volunteers described the experience in greater detail verbally than subsequently on paper.

6. RESPONSES AND ANALYSIS

The responses were tabulated for analysis to identify key themes.

Of interest were the variations in descriptive language across such a mixture of untrained listeners varying in age, gender, expertise and listening ability. A broad synonymic approach was taken, whereby terms were loosely grouped to form themes. So, for instance, the category “weird” included terms such as “eerie”, “strange” and “unusual”.

Theme	Descriptors in Class
Positive	Nice, Incredible, Amazing, Awesome, Excellent, Loved, Good, Enjoyed, Cool, Wonderful, Extraordinary, Impressive, Effective
Negative	Muddy, Muffled, Lacking, Limited, Quiet, Dull, Distortion
Hearing Loss	Hearing Loss
Spatial	Spatial, Surround, 3D, Virtual Reality, Image location, Movement, Image positioning, 360 soundfield, External
Clarity	Clarity, Clear, Crisp, Pure
Interesting	Interesting, Fascinating
Weird	Weird, Unusual, Surreal, Strange, Uncanny, Ethereal, Eerie, Bizarre
Vibrations	Vibrations, Tickling, Tickling
Feel	Feel, Felt, Feeling, Natural, Sensorial
External	Distant, Immersive, Overhead, Above, Around, Spacious, Outside
Headphones	Headphones

Figure 3: Main themes and descriptive terms

The emergent key themes in aggregated comments were: “positive” [77%] (expressed as having enjoyed the experience), “spatial” [38%] (including surround/surrounded, spacious, distant, immersive, above etc.), “interesting” [38% (including “fascinating”, “amazing” “incredible” etc.,), “weird” [23%] (including “eerie”, “strange” “unusual”), “vibrations” [24%] (expressed directly as vibrating, vibrations) , “clarity” [22%] (clear, pure,), “feelings” [28%] (distinct from vibrations, such as “felt very pleasant” “felt dreamlike” “felt like I was in the soundscape”), and “negative” [19%] (expressed as “not clear enough” or “couldn’t hear the bass”). A complication arose in the overlap of the positive and negative categories, 10**%** of respondents gave comments that included both. 14% of comments were classed as “neutral”.

6.1 Participant comment samples

1) Male age 30, Sound Engineer, non-musician.

“Very surreal distant sounding. Passing sounds such as the train and plane felt closer and move forward. The higher sounds such as water felt harder to make out. Fidelity sometimes felt lost when many sounds were overlapped. As strange as it sounds it was like a memory or dream of a sound.”

Recorded classes for comment 1:

Positive; Negative; Spatial; Surround; Feel; External; Weird.

2) Female age 36, Stage Manager, non-musician.

“Although the sound was still 'one sided ' to a certain degree I felt for the first time that I was immersed in a soundscape and that my hearing loss was not making me lose out on part of the effect. The train in particular really felt 360, especially with the chin transducer on my right cheek bone.”

Recorded classes for comment 2:

Positive; Hearing Loss; Spatial; External; Feel.

3) Male age 62, Concert Producer, Musician

“Sounded slightly “muffled” some spatial “separation” but not dramatic”

Recorded classes for comment 3:

Negative; Spatial.

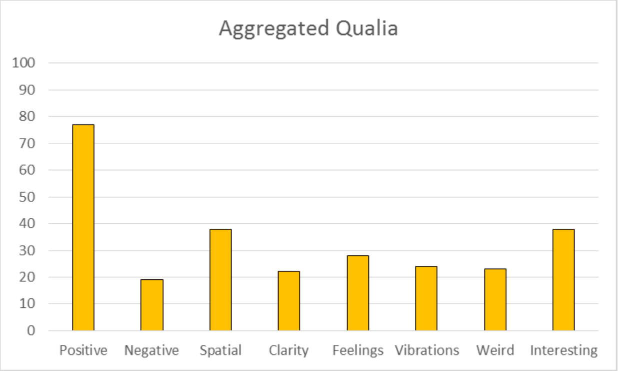


Figure 3. Aggregated Qualia

Notably, the degree of emphasis placed on each attribute (for instance, “quite spacious” or “very spacious”) was not distinguished here.

6.2 Co-occurring themes

We then mapped each attribute class against “positive” to find what it was about the experience that people found rewarding.

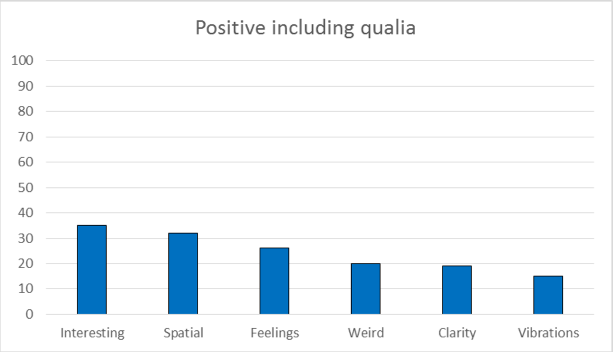


Figure 4: Qualia included with positive comments.

We found that the attribute class that mapped most strongly to positive expressions was the “interesting” category; 35% of comments included interesting and positive descriptors. This was followed by the spatial class – 38% had used spatial terms; 32% had used spatial *and* positive terms. Those that referred to the way they felt about the experience also correlated highly with positive – 26% featured positive *and* feelings. Clarity was referred to in conjunction with positive comments in 19%, "vibrations" were mentioned along with positive comments in 15% of cases.

7. DISCUSSION

Results may indicate that tissue conduction is of more utility to some than others; variations in comments might also indicate variances in biomechanical and/or neurological auditory processing.

Some volunteers (3%) reported some degree of bilateral or unilateral hearing deficit, but nevertheless reported in spatial and positive terms. Some others (2%) reported spatial anomalies that might indicate a degree of unilateral deficit ("the sound field sounded shifted to the left") but it was not within our experimental purview to comment or diagnose. Likewise, some that had used very positive and clarity terms may actually have been observing differences between their normal air-conducted hearing and this experience.

The surprising concomitance of reports of vibrations (which we might have thought was an undesirable percept) and positive comments prompts us to speculate that program-material modulated haptic input can contribute to the experience.

Notably, in the case of the "weird category (23%), weird and positive comments appeared in conjunction in 11% of all comments; there may be an overlap in the "weird" and the "interesting" categories, depending on individuals' use of language. It does appear that the novelty of the experience may be conflated with positive reports, and this in itself does not imply improvement in informational throughput.

8. CONCLUSIONS AND FURTHER WORK

This work is the early stage of investigation as to what might provide valuable experiences in tissue conduction, for whom.

Early indications are that the qualia associated with this kind of spatial experience may be similar but not identical to those for binaural presentations. Hence more structured methodologies should not precisely mimic those for air-conducted hearing.

At this stage of prototypical development, display of spatial parameters cannot be deemed accurate. Precision localization (of sources) in terms of azimuth, elevation and source-perceiver range is currently infeasible. Nevertheless, the fact that some degree of externalization and sense of spaciousness were alluded to in listeners' observations, is of interest. Research into refined processing of the signal set dedicated to tissue conduction is indicated. Ambisonic encoding has been used as a methodological convenience; its advantages for some attributes (such as ambient spaciousness) might not be matched for others (such precision localization). Different spatial audio attributes may be favored in different applications, of which personal music listening is only one. Similarly, it may be that a single spatial music encoding regime will not be appropriate for all listeners.

This work has enabled us to identify the following development areas for future research:

Technological: improved signal processing, improved transduction, improved apparatus comfort, developments in multimodal stimuli.

Methodological: Precise characterization of listener hearing capabilities, investigation of training periods and of individual preferences for encoding. Parameterization of qualia for spatial music listening.

Possible benefits of competent spatial tissue-conduction apparatus include:

- Enhanced quality of life for those with conductive hearing loss, through access to personal music listening.
- Augmented private perception where unimpeded air-conducted hearing is required.
- Diagnostic procedures to identify and isolate conductive hearing loss components.
- Improved methodologies for the investigation of mutimodally-augmented perception.

9. REFERENCES

[1] G. Martin, W. Woszczyk, J. Corey and R. Quesnel "Controlling Phantom Image Focus in a Multichannel Reproduction System" in Proc. 107th Conference of the Audio Engineering Society (AES 107), New York, 1999

[2] J. Blauert, W. Lindemann, "Auditory spaciousness: Some further psychoacoustic analyses," in J. Acoustical Society of America (JASA), 1986, 80(2):533-42

[3] T. Neher, T. Brookes, F. Rumsey, "Unidimensional simulation of the spatial attribute 'ensemble depth' for training purposes. Part 1: Pilot study into early reflection pattern characteristics," in: Proc. Audio Engineering Society 24th International Conference on multichannel Audio (AES 24), 2003, pp. 123-137

[4] R. Einhorn "Observations From a Musician With Hearing Loss" in Trends Amplif. 2012 Sep; 16(3): 179–182. doi: 10.1177/1084713812468513

[5] R. Guski, "Psychological methods for evaluating sound quality and assessing acoustic information" *Acustica* 83, 1997 pp 765-774

[6] N. Ford, F. Rumsey, B. de Bruyn, "Graphical Elicitation Techniques for Subjective Assessment of the Spatial Attributes of Loudspeaker Reproduction – A Pilot Investigation" in proc. Audio Engineering Society (AES) 110th Convention, Amsterdam, 2001

[7] World Health Organisation "WHO global estimates on prevalence of hearing loss", Report on WHO website, 2012,

http://www.who.int/pbd/deafness/WHO_GE_HL.pdf?ua=1 last accessed Feb 2017

[8] World Health Organisation, “Hearing loss in persons 65 years and older based on WHO global estimates on prevalence of hearing loss; Mortality and Burden of Diseases and Prevention of Blindness and Deafness” report on WHO website, 2012, http://www.who.int/pbd/deafness/news/GE_65years.pdf?ua=1 Last accessed Feb2017

[9] R.R Ruby “Conductive hearing loss in the elderly” in J. Otolaryngology, 1986, 15(4): pp. 245-247

[10] A. Tajadura-Jiménez, A. Väljamäe, N. Kitagawa, and H. Ho, “Whole-Body Vibration Influences Sound Localisation in the Median Plane”, Proceedings of the 10th Annual International Workshop on Presence, Barcelona, Spain, Oct. 2007.

[11] M. Gerzon, “What's wrong with quadraphonics?” Studio Sound, 16(5), 1974pp. 50–56.

[12] R. Zvonar, “A history of spatial music: Historical antecedents from renaissance antiphony to strings in the wings,” in eContact!: The online journal of the Canadian Electroacoustic Community (CEC), 7(4), 2005, [Last accessed October 2015]

[13] P.P. Lennox, P. P. “The philosophy of perception in artificial auditory environments: Spatial sound and music,” PhD Dissertation, Dept. Music, Univ. York, UK. 2005

[14] S. Gould and S. Vrba, “Exaptation—a missing term in the science of form” Paleobiology 1982 8: 4-15.

[15] W. Snow, (1953). “Basic principles of stereophonic sound,” J. Society of Motion Picture & Television Engineers (SMPTE), 1953, 61, pp. 567–587.

[16] D. Griesinger, “General overview of spatial impression, envelopment, localization and Externalization,” In Proc. of the 15th International Conference of the AES on small room acoustics (AES 15), Denmark, 1998, pp. 136–149

[17] E. Bates “The Composition and Performance of Spatial Music” – PhD dissertation. Dept. Music, and Dept. Elect. Eng., Trinity College Dublin, Republic of Ireland, 2009

[18] R. Stanley, & B.N. Walker, “Lateralization of sounds using bone-conduction headsets”, Proceedings of the Annual Meeting of the Human Factors and Ergonomics Society (HFES) San Francisco, CA. 2006 pp. 1571-1575

[19] J.A. MacDonald, P.P. Henry, & T.R. Letowski, “Spatial audio through a bone conduction interface”, International Journal of Audiology. 2006 45, pp. 595-599.

[20] S. Stenfelt, & M. Zeitooni, “Binaural hearing ability with mastoid applied bilateral bone conduction stimulation in normal hearing subjects”, Journal of Acoustical Society of America, 2013 (134), 1, 481-493.

[21] W. A. Yost, “Lateral position of sinusoids presented with intensitive and temporal differences”, Journal of the Acoustical Society of America, 1981 70:397–409, 1981.

[22] R. O. Duda, “Elevation dependence of the interaural transfer function,” in Binaural and spatial hearing in real and virtual environments (R. H. Gilkey and T. R. Anderson, eds.), Mahwah, New Jersey: Lawrence Erlbaum Associates, 1997 ch. 3, pp. 49– 75.

[23] B. C. J. Moore, “An introduction to the psychology of hearing”, Academic Press, London, fifth edition, 2003.

[24] M.A. Meredith, & B.E. Stein, B. E, “Visual, auditory, and somatosensory convergence on cells in superior colliculus results in multisensory integration”, Journal of Neurophysiology Published 1 September 1986 Vol. 56 no. 3, 640-662 DOI:

[25] C. Pantev, A. Wollbrink, L.E. Roberts, A. Engelien, & B. Lu¨tkenho¨ner, “Short-term plasticity of the human auditory cortex”, Brain Research 842 1999 192–199.

[26] A. Good, M.J. Reed, & F.A. Russo, “Compensatory Plasticity in the Deaf Brain: Effects on Perception of Music”, Brain Sci. 2014, 4, 560-574; doi:10.3390/brainsci4040560

EXPLORING SOUND-MOTION TEXTURES IN DRUM SET PERFORMANCE

Rolf Inge Godøy
University of Oslo
r.i.godoy@imv.uio.no

Minho Song
University of Oslo
minho.song@imv.uio.no

Sofia Dahl
Aalborg University Copenhagen
sof@create.aau.dk

ABSTRACT

A musical texture, be that of an ensemble or of a solo instrumentalist, may be perceived as combinations of both simultaneous and sequential sound events. However, we believe that also sensations of the corresponding sound-producing events (e.g. hitting, stroking, bowing, blowing) contribute to our perceptions of musical textures. Musical textures could thus be understood as multimodal, with features of both sound and motion, hence the idea here of *sound-motion textures* in music. The study of such multimodal sound-motion textures will necessitate collecting and analyzing data of both the produced sound and of the sound-producing body motion, thus entailing a number of methodological challenges. In our current work on sound-motion textures in music, we focus on short and idiomatic figures for different instruments (e.g. ornaments on various instruments), and in this paper, we present some ideas, challenges, and findings on typical sound-motion textures in drum set performance. Drum set performance is particularly interesting because the often very complex textures are produced by one single performer, entailing a number of issues of human motion and motor control.

1. INTRODUCTION

Our subjective experiences of any drum set performance may be characterized as that of *sound-motion textures*, meaning holistic perception of time-limited sequences of sound and body motion events, be that only of a single measure, or of a fragment of several measures, or of a whole work of music. Although the term *groove* is often used to denote various drum set patterns, we prefer to use the generic term *texture* to signify a more general notion of sound-motion patterns, so that *texture* may also denote all kinds of figures, e.g. such as ornaments.

The expression *sound-motion textures* also implies that we have a multimodal approach to rhythm perception and production, meaning that we understand the experience of rhythm as composite, as including both sound and motion features.

This in turn means that we need to consider body motion idiosyncrasies and constraints, as well as possibilities and constraints of the musical instruments involved. If we as a thought experiment imagine a drum set playing machine with solenoid activated mallets, we would have no biological constraints on the sound production. Such a machine could conceivably produce some sound-motion textures resembling that of a human drummer, but obviously without the motion constraints of the human body. So, if we could see this machine in action, it would probably also be at odds with our perceptual schemata of drum set music. In line with a so-called *motor theory perspective* on perception, we believe that the choreography of sound-producing body motion is an integral part of musical experience, see e.g. various contributions in [1].

The central idea here is that drum set sound-motion is produced by one performer, and that we at times may see an extraordinary virtuosity by drummers in handling several instruments. That such highly complex textures can be produced by one person, could lead us to take an *egocentric* perspective here, meaning a body-centered frame of reference for all that goes on [2].

Such an egocentric perspective may be extended to the perception of drum set textures, as demonstrated by some cases of *beatboxing*: reproducing a complex instrumental texture by the vocal apparatus could also suggest that we (variably so) may have a capacity for perceiving and imagining even quite complex sound-motion textures. But notably so, beatboxing is basically 'monophonic' because of the restriction of our vocal apparatus, yet this apparatus is capable of a sequential production of what may suggest a 'polyphonic' texture, similar to how polyrhythmic sequences may be conceived of, and controlled, as single sequential patterns [3], and all the time from a similar body-centered perspective.

The aim of the present paper is to give an overview of how we try to explore sound-motion textures in drum set performance, and in particular try to understand more of how a single performer can generate so complex, and also compelling, textures. With this aim in mind, we shall first consider some issues of sound-motion in view of multimodality, moving on to challenges of documenting sound-motion, followed by a presentation of various constraints of the sound producing body motion and the instruments used. We shall also present how we try to integrate our data into a coherent model of motion hierarchies and into what we call *sound-motion objects*, before listing what we see as some main challenges and directions for future work.

2. MULTIMODALITY

The basic tenet of our work is that sound-motion textures are multimodal, hence that we need to consider the perception of drum set performances as involving both sound and body motion. The sound-motion relationship may at times be so close that it is difficult to differentiate clearly between these two, i.e. difficult to distinguish between what is a sound event pattern and what is a motion event pattern in a textural excerpt. We may suspect that this interweaving of sound and motion also extends to aesthetic and affective features, such that e.g. subjective impressions of energy and gait in musical experiences combine sound and body motion sensations.

We here have the non-trivial task of trying to differentiate what goes on in complex drum set textures. First of all, this may be regarded as a more conceptual or ontological issue of trying to distinguish the ingredients in the multimodal mix. We should ask questions about what constitutes a sound event, and what constitutes a motion event. And furthermore, we should ask questions of what are the criteria for fusion of small-scale events into larger-scale events, e.g. between what is perceived as individual impacts, and what is perceived as a more fused stream of onsets e.g. as in a fill. Also, it will be useful to try to distinguish between what is strictly sound-producing body motion and what may be more sound-facilitating or even sound-accompanying motion of a drummer, e.g. recognizing what may be more communicative or theatrical body motion in a drum set performance [1].

Additionally, we need to take into consideration the acoustic features of the instruments as well as the logistics of the drum set instrument layout, as this is the spatial framework for the choreography of sound-producing body motion.

The next challenge will be to have a reasonably good understanding of what goes on in the production of these (at times quite complex) sound-motion textures. This will involve recording the body motion of the drummer, first of all the hands and arms, but also the whole torso and the head, as well as the feet. The use of both feet in drum sets necessitates balancing the body in the seated position so as to enable free motion of all four limbs (right and left hands/arms, right and left feet). Besides entailing various biomechanical constraints, this use of all four limbs will also needless to say be very demanding in terms of motor control. In addition to requiring extended practice, skilled drum set performance evidently also requires some efficient hierarchical and anticipatory control schemes.

Exploring the multimodal sound-motion relationships necessitates that we try to find out more about what goes on at the different timescales: clearly, we have the singular mallet-instrument impact-points and the corresponding sound onsets, but we also have the motion trajectories to and from such impact-points, trajectories that are both extended in time and usually involve more components of the effector apparatus, e.g. mallet-hand-wrist-lower arm-elbow-shoulder-torso.

In sum, this multimodal approach means that we need to consider chunks of sound-motion as holistic and coherent entities, as what we later shall call *sound-motion objects*, and try to understand how these sound-motion objects have new and emergent features not present at the timescales of individual impact-points.

3. DOCUMENTING SOUND-MOTION

In our explorations of drum set sound-motion textures, we need to collect data on both the sound and the sound-producing body motion, as well as to make extensive correlations between these two domains.

Starting with the instruments, we need to consider how the instruments work, i.e. know about their mode of excitation and resonant features, and importantly, the logistics of their positioning. Needless to say, one of the key elements of drum set performance is the instrument layout, enabling the performer with minimum effort, and at times also great speed, to alternate between the different instruments, cf. the layout of instruments depicted in Figure 1.

In earlier recordings, an acoustical drum set was used, but this required the use of so-called *active markers* for the infrared motion capture system, because the shiny metal parts of the instruments totally confused the tracking system so that passive markers could not be used. The use of active markers turned out to be both cumbersome (requiring wiring of the mallets) and not sufficiently reliable, making us use a MIDI drum set (Roland TD20) in subsequent recording sessions. Using MIDI drums made the use of ordinary passive markers possible, as well as gave us precise onset timing from the MIDI data. In our subjective opinion, the sound quality of the MIDI instruments was satisfactory, i.e. seemed to well reflect the different playing modes as well as the resonant instrument properties, and produced satisfactory contextual overlap for sound coarticulation (more on coarticulation below).

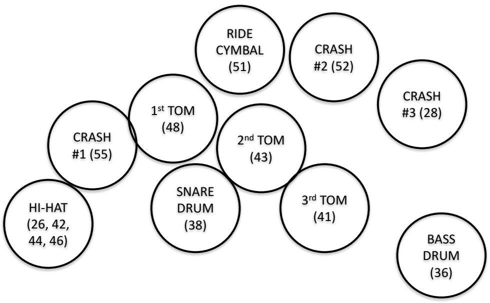


Figure 1. Layout of the MIDI drum set used in the recording sessions (the bass drum is actually placed below the 2nd and 3rd toms).

The data reported in this paper is based on recordings with a professional drum set player across one final session carried out after several pilot sessions over a period of one and a half years. During all these sessions, the drummer would perform a series of grooves and fills at different tempi, and often with multiple repetitions enabling future studies of motion invariants across repetitions.

For capturing the drummer's motion, we used an infrared motion capture system (a 13-camera Oqus Qualisys system, see http://www.qualisys.com) recording at 240Hz, with the drummer wearing a suit and cap equipped with reflective markers, and using drum sticks likewise equipped with reflective markers. Sound and MIDI data was recorded separately, and for reference, the recording sessions were also documented on video.

Further processing and graphing was done in *Matlab*, and there was not applied any smoothing here except for on the data reported in the middle plot of Figure 4, where a 5-points moving average for elbows, and a 17-points moving averages for wrists, was used.

Although infrared marker-based motion capture systems presently seem to be the most versatile method for collecting data on sound-producing body motion, it does also entail some serious challenges, first of all with the at times very rapid motion in expert drumming. Besides the mentioned issue of unwanted reflections, there is also the issue of dropouts, and most seriously, of not being able to completely capture what goes on even with very high frame rates (sample rates). As shown in [4], it may be necessary to have very high frame rates when studying detailed and fast motion.

Furthermore, capturing the precise moment of impact is not possible even with very high frame rates (in theory, approaching infinitely high sampling rates), so the use of a MIDI drum set is a practical solution in this regard also. As can be seen in Figure 2, there is a discontinuity in the motion trajectories around the point of impact, i.e. at the minima of the motion trajectories. That said, the point of impact may of course be estimated from the position data of the recorded motion frames by various means, such as by *functional data analysis* (see e.g. [5]).

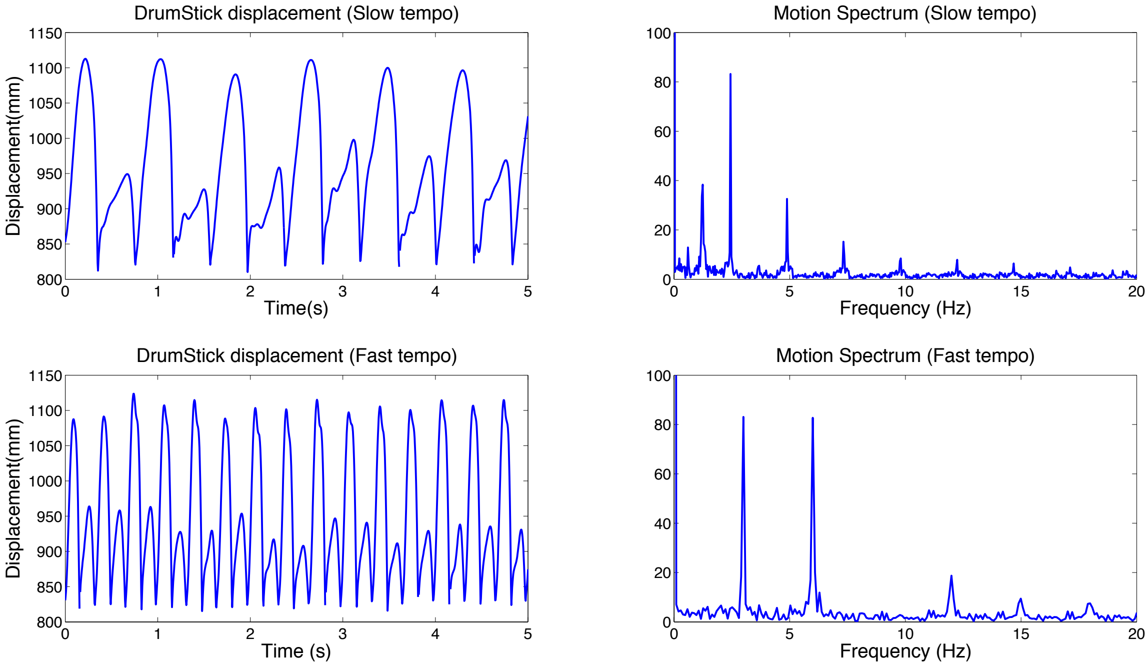


Figure 2. Two motion tracking examples using passive markers attached to a drumstick recorded at 240Hz. The two examples had different tempi (*slow, fast*) with mean velocities of 1448 mm/s and 3564 mm/s. Their peak velocities were 10096 mm/s and 11666 mm/s. Although the mean velocity of the fast motion was almost 2.5 times higher than that of the slow motion, we can see that peak velocities were almost the same. It means that the end velocities of single strokes do not vary significantly with respect to tempo. Also, every time when the marker reaches the lowest position (point of collision), we could observe discontinuities in the trajectories for both cases. It is interesting to note that the dominant frequency of the motion was less than 10 Hz, (see the right column), but the frame rate of 240Hz was not enough to capture the motion.

With our focus on sound-motion textures, we have the additional challenges of not only finding out what goes on in singular strokes, but also in an ensemble of markers to capture the overall motion involved in more composite textures.

The motion capture data will even with much care taken to avoid the problems presented above, usually have some need for preprocessing, typically removing spurious reflections, gap-filling dropouts, assigning markers to points on the body, and applying skeleton models. Usually, there will be a need for further processing that can result in representations as graphics and animations of the motion trajectories, as well as of derivatives, in particular of velocity, acceleration, and jerk, as well as more cumulative measures, such as quantity of motion. Processing is typically done in *Matlab* and associated toolboxes [6] or other software, e.g. *Mokka*, and the hope is to develop means for more synoptic overviews as suggested by Figure 4.

Further on, there are challenges of detecting and representing musically salient features from the continuous stream of motion data, challenges that border on to issues of motor control and perception, such as:

- Accurate onset detection, both of sound and motion
- Challenges of representing multiple trajectories (i.e. markers) in graphs, see e.g. Figures 4 and 5 below
- Discerning and understanding variability vs. constancy in trajectories, cf. the point made by Nikolai Bernstein as "An instance of repetition without repetition", meaning that we very often find approximate (i.e. non-exact) similarities between motion chunks in human behavior [7, p. 1726].

4. CONSTRAINTS

Needless to say, there are a number of constraints on non-electronic sound production in music. First of all, there are constraints of the instruments, such as their repertoire of possible output sounds and modes of performance. This includes various constraints such as the required force of excitation, of reverberation time, the need for damping, and also the need for positioning the effectors prior to impacts, in turn entailing important logistic elements in the layout of the instruments in the drum set that have consequences for the sound-producing body motion:

- All instruments must be reached from the sitting position.
- The drummer must be seated so that both the right and left foot can be used independently.

And there are of course many constraints on human sound-producing body motion: limitations on speed and amplitude of motion, on endurance, the need for rests and need for minimizing effort in order to be able to sustain such motion for extended periods of time and to avoid strain injury. In addition to such mostly biomechanical constraints, there are motor control constraints such as need for hierarchical and anticipatory control. In sum, we have the following main constraints on the production of sound-motion textures:

- There are basic motion categories that seem to be grounded in a combination of biomechanical and motor control constraints: *impulsive*, also sometimes referred to as *ballistic*, meaning a very rapid and brief muscle contraction followed by relaxation, such as in hitting; *iterative*, meaning a rapid back and forth motion such as in shaking the wrist or in rolling the hand; *sustained*, meaning continuous motion and mostly continuous effort, such as in stroking, but this category is not so common in drumming (found in scraping the cymbal, brushes on drums). Such a *motion typology* is clearly at work in shaping musical sound, hence, the labels may equally well apply to sound [8].
- All motion takes time, i.e. instantaneous displacement is not possible for the human body, and this in turn will lead to so-called *temporal coarticulation* in relation to sound onset events, i.e. that there is a contextual smearing of motion from event to event [9].
- Motion of end effectors, e.g. hands or feet, may also recruit motion of more proximal effectors, e.g. arms, shoulder, torso, and even whole body. This leads to so-called *spatial coarticulation*, meaning the recruitment of more effector elements than the end-point effectors, depending on the specific task [9].
- Coarticulation is an emergent phenomenon, cf. [10, p. 592]: "a control mechanism in which the different motor elements are not simply linked together in the correct sequence, but are also tuned individually and linked synergistically based on the final goal, with coarticulation observed as an emergent phenomenon."
- And furthermore, as argued in [10, p. 592], coarticulation is related to *chunking*, i.e. to the "…the integration of independent motor elements into a single unit. With chunking, there can be an increase of coarticulation and reduced cognitive demands because less elements are organized for a given motor goal"... "Chunking is also a critical element for automatization to emerge."
- *Left hand/right hand alternations* is a way around the constraints of speed, allowing for extremely rapid figures, similar to tremolo alternations in one hand figures on various other percussion instruments and on keyboards, and actually related to the back-and-forth shake motion of wrist tremolo in string instruments and triangle (or other confine-based instruments such as tubular bells).
- Phase relationships of left-right, cf. [2], suggesting different levels of stability depending on phase relationships.

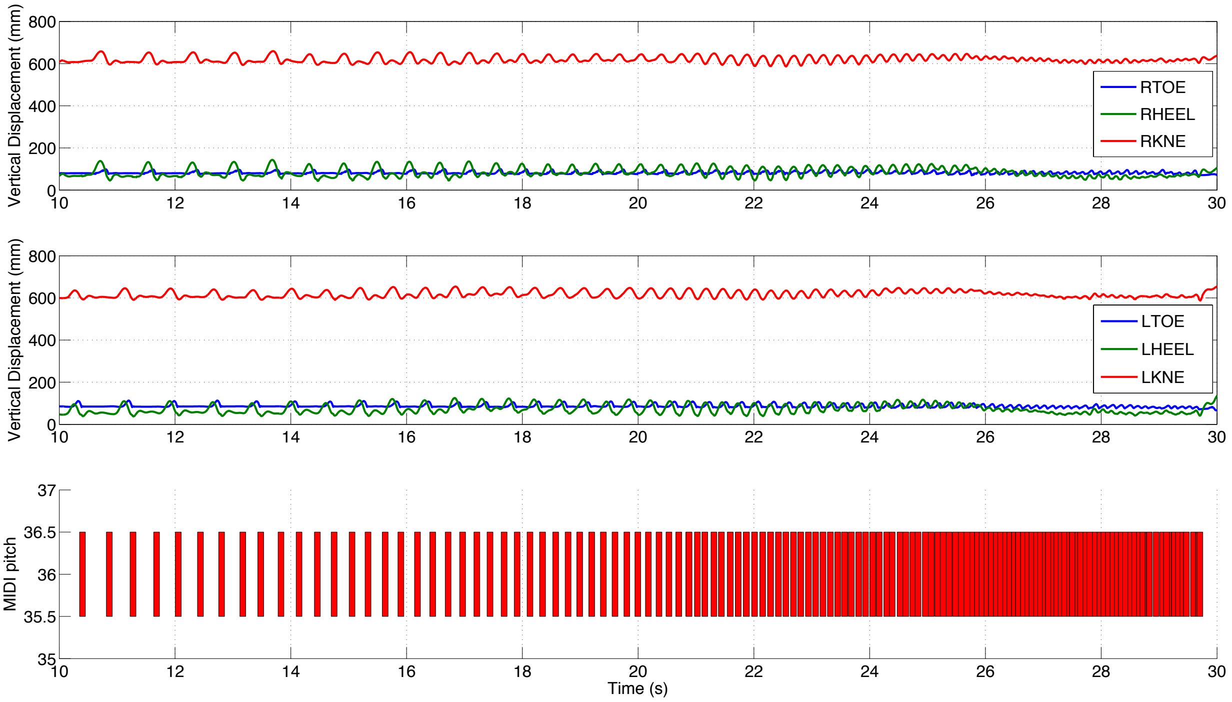


Figure 3. The increasing density of motion events leads to phase-transition, similarly to how event proximity may lead to coarticulatory fusion. Here we see how a repeated and accelerated right foot – left foot bass drum pedal motion leads to a phase-transition after approximately 20 seconds with the disappearance of the dip after the impact as well as a diminishing of the amplitude of motion, something that seems reasonable given the speed constraints of the human body.

In addition to fusion by coarticulation, we can also find instances of fusion by so-called *phase-transitions* in the sound-producing motion [9]. Phase-transition means that a series of otherwise independent events may fuse into what is perceived as a continuous stream because of increased event density, and conversely, a continuous stream of events may change into a series of distinct events with a decrease in event density. The transition from single strokes to a drum roll, or from single tones to a tremolo or trill, are common examples of this. In Figure 3, we see an example of such phase transition in the bass drum foot pedals of the drum set. With an acceleration, we see the change in the motion shape, and at the fastest tempo, also a decrease in the amplitude of the motion.

With both coarticulation and phase-transition we have cases of fusion, what we in general refer to as *chunking*, that seem to emerge from a combination of biomechanical and motor control constraints (for the moment, it seems not possible to say exactly what is due to what), i.e. gives us a bottom-up factor in the shaping of sound-motion textures.

Additionally, it seems that skill in performance is also about minimization of effort, and that virtuoso instrument performance in general gives an overall impression of ease, something that seems related to the exploitation of motion hierarchies.

5. HIERARCHIES

Using both hands and both feet in at times very dense and rhythmically complex drum set passages, is clearly demanding both in terms of effort and motor control. From what is known of human motor control (see e.g. [11]), such skillful behavior necessitates that the motion be partially automatic, i.e. need some kind of control hierarchy. Likewise, the layout of the human body will inherently result in some motion hierarchies, i.e. the fingers, attached to hands, attached to arms, shoulders, torso, and whole body, hence, we have hierarchies both in the spatial and in the temporal domain, similar to having coarticulation in both the spatial and temporal domain.

Spatial hierarchies are evident in task-dependent effector mobilization, e.g. for soft and slow textures, there is usually only a need for the most distal effectors, e.g. hand and wrist, whereas in more loud and fast and dense textures, it may be necessary to recruit more proximal effectors, e.g. elbow, shoulder, and even torso.

As for temporal hierarchies, these are probably the most crucial for achieving high levels of performance skill. Such hierarchies are again clearly dependent on extensive practice, practice resulting in exploitation of various synergies (i.e. cooperation of muscle groups) and hierarchies, i.e. top-down 'commands' at intermittent points in time [12]. Hierarchies then go together with a basic discontinuity in human motor control, and is associated with anticipatory motion [13], as well as with high levels of preprograming, i.e. with what has been called *action gestalts* [14]. The turn towards discontinuity in human motor control in some recent research is interesting for musical applications, because the demands for very rapid motion would indeed go better with intermittent motor control than with more traditional models of so-called 'closed loop', i.e. continuous, control because of its slowness [15].

More specifically, we believe there is evidence for how seemingly complex bimanual tasks can be carried out by a human performer provided a hierarchical approach. As suggested in [3], a polyrhythmic passage may be thought of as a singular rhythmic pattern, e.g. a pattern of 3 against 4 quarter tones may be thought of as a series of one punctured eight note, one sixteenth note, two eight notes, one sixteenth note, and one punctured eight note. This very simple example may be seen as what in [2, p. 103] is called a case of when "a dual task becomes a single task", and which the authors believe works because "the spatial patterns produced by the 2 hands form a geometric arrangement that can be conceptualized as a unified representation."

Another aspect of facilitating hierarchies is that of alternations between effectors and/or effector positions. It is well known that for instance a tremolo on a keyboard or with two mallets on a percussion instrument is much easier using a left-right-left-right-etc. hand rolling motion than a series of up-down motions of the whole hand. And likewise, in the vocal apparatus, it is possible to make a very fast articulation of saying *ta-ka-ta-ka-tam*, whereas saying *ta-ta-ta-ta-tam* is only possible at a slower rate. In the drum set, this alternation between effectors can lead to very complex passages, and although the motor control factors enabling such virtuosity remains to be explored, the idea of some kind of hierarchical and intermittent control scheme does seem quite plausible.

On the perceptual side, there are probably also *sound hierarchies*, i.e. that there is the *reverberation* of the instruments contributing to the basic *smearing*, the *phase-transition* leading to *fusion* (e.g. iterative rather than impulsive sound) and that the *coarticulatory inclusion* of successive sounds (sometimes also simultaneous sounds, e.g. bass drum + the other drums) contribute to holistically experienced chunks of sound-motion textures.

6. SOUND-MOTION OBJECTS

From a combined sound and motion perspective, it seems clear that sound-motion textures in drum set performance are both perceived, and in their production, conceived, as chunks, gestalts, or holistic entities, entities that may vary in content, e.g. between single slow strokes and ferocious fills, but typically experienced as units, as what we like to call *sound-motion objects*. The main reasons for the idea of sound-motion objects are the following:

- *Intermittent control*, based on the need for anticipation and on the possibilities of action gestalts in human motor control [12, 14, 15].
- *Intermittent energy infusion*, in particular with ballistic motion, i.e. biomechanics in combination with instrument physics (requiring impact excitation), and biomechanical demands for rests [9].
- *Coarticulation*, which is a constraint-based phenomenon that results in object level fusion, also related to *phase-transition* in cases of repetitive motion [9].
- *Chunk timescale musically most salient*, meaning that the timescale of approximately between 0.5 and 5 seconds is the most salient for most musical features, cf. the work of Pierre Schaeffer on sonic objects and similar work on timescales in music [8].

The notion of sound-motion objects is then not only in accordance with some basic features of perception and motor control, but also advantageous in view of various music theory and/or analytic perspectives. With such a central perspective of sound-motion objects, we believe we are in a better position to interpret the various sound and motion data that we collect in our research:

- Regard sound-motion textures as holistic objects, often repeated, often with variations, but variation-tolerant in retaining identity, cf. Bernstein's idea of "repetition without repetition" mentioned above [7].
- Also in cases of gradually increasing density of fills, e.g. from none by way of various upbeats and syncopations to very dense and energetic fills, a hierarchical scheme of salient points in the music, i.e. what could be called "anchoring" [7], seems to be at work.
- Focusing on sound-motion objects could also be useful for studying the aesthetic and affective aspects of drum set music, i.e. for studying how the various motion features, in particular the quantity of motion, the velocity, and jerk (derivative of velocity) may contribute to the overall experience of the music.

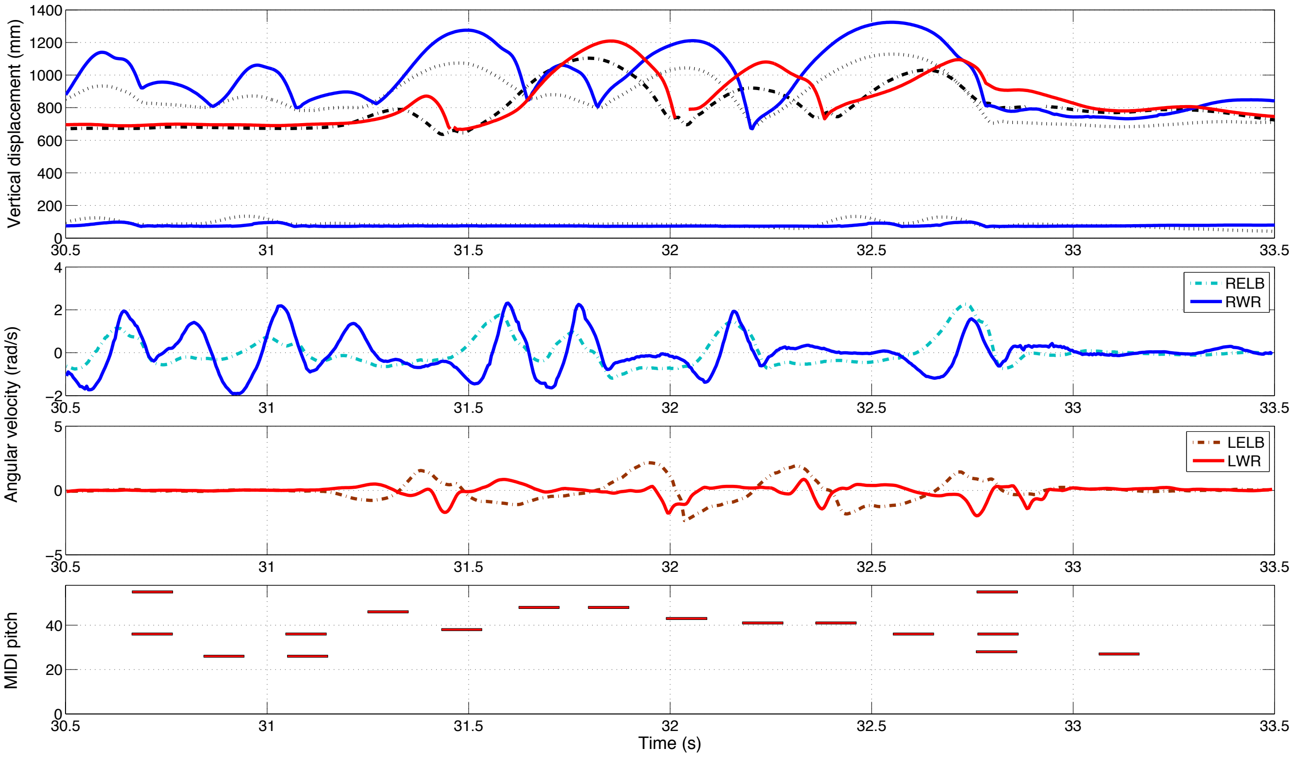


Figure 4. Motion of markers and MIDI onsets vs. time during the end fill after playing a steady slow groove. Top panel: The vertical displacement of markers on the player's right ankle (lower dotted line), right toe (lower blue line), right hand (upper dotted trace), left hand (dash-dotted line), right stick (upper blue line) and left stick (red line). As can be seen, the traces of the hand markers tend to precede the stick movements slightly, but otherwise maintain similar shape albeit with lower amplitude. The ankle marker is also preceding the toe marker, but this ankle movement is larger. Middle panels: the angular velocities of the right and left wrists and elbows. Bottom panel: MIDI onset times and drum info.

In our studies of sound-motion textures, we have seen how rather demanding passages may be performed seemingly with ease and elegance, and after quite extensive reviews of our recordings, we believe that the principles of sound-motion objects are crucial for both the performer and for the listener. As an example of this, consider the three-second fill passage from the end of a slow eight measure groove in Figure 4 and 5. Subjectively, the passage is a different object from the main pattern of the slow groove, but it is right on the beat of the groove. In the course of the fill, the drummer also makes a right torso rotation, probably in order to have the effectors in the optimal position for the final impacts of the fill.

In Figure 4, we see the motion of the hands and the feet in this three-second fill, as well as a zoomed in image of the angular velocity of the right and left elbows and wrists, indicative of the motion to the impact-points in the fill, and the MIDI onset points of these impact-points.

In Figure 5, we see a 3-D rendering of the right and left drumsticks and hands in the same three-second groove, demonstrating the more large-scale motion also involved in this fill.

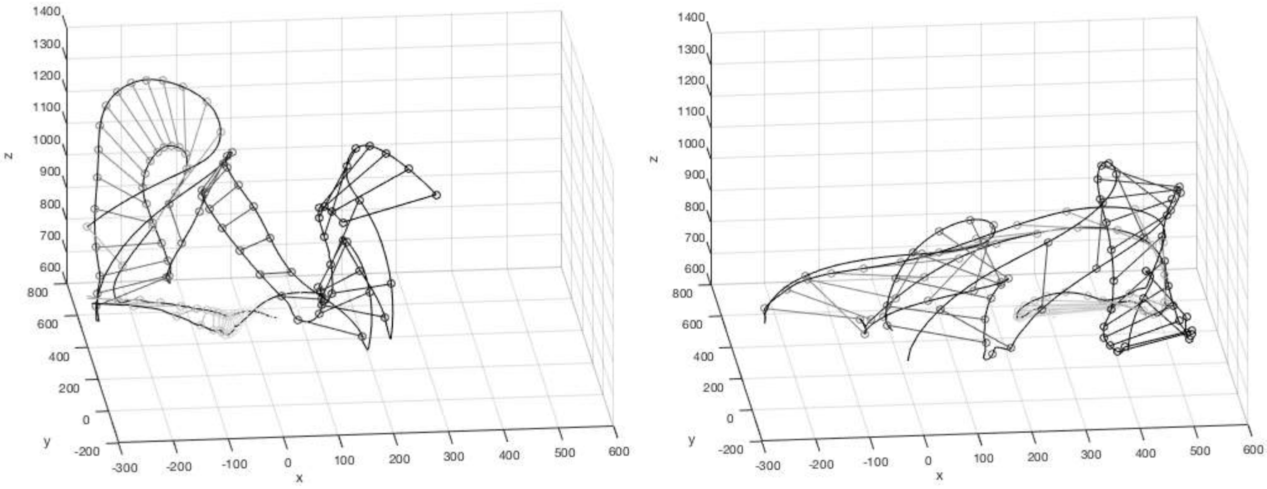


Figure 5. The movement for markers attached on the drummer's right/left hand and drumsticks during a final fill depicted in Figure 4 (from 31.4s to 33.5s). Every 8th sample point is marked with a circle and a line has been drawn between the marker at the stick and the hand, connecting the two. The movement start in the (upper) right hand side of the figures, where the right hand and stick are lifted in preparation for the first stroke in the fill while the left hand is at a lower amplitude (compare Figure 4). Thereafter the arms move to other drums in the left of the figure, and, finally, back to the snare.

7. CONCLUSIONS AND FURTHER WORK

Clearly, drum set music is a multimodal art with its combination of sound and motion, the latter in turn a combination of effort-related, visual, proprioceptive, etc. sensations. Sound and motion are so intertwined in both production and perception that it may be difficult to separate these two main elements, and should be a good reason for using the term *sound-motion textures*.

Our studies of drum set sound-motion textures have a number of issues in common with sound-motion textures on other instruments, and in particular rapid textures such as typically encountered in ornaments. We believe these rapid figures are important to study because they could shed light on how skilled musical performance can work around various constraints by anticipatory cognition, and also shed light on how small-scale sonic events fuse into sound-motion objects with emergent features not found on the small-scale level. In view of these aims, we hope to in the future be able to work on following topics:

- Enhanced motion capture methods that give us more precise recordings of motion trajectories.
- Motion capture recordings supplemented with so-called *electromyographic* (EMG) recordings of muscle activity of the effectors, giving us more information about both the level of effort and the timing of muscle contraction in relation to the visible motion as well as the auditory output, hence also shed light on issues of chunking.
- Continue our studies of sound-motion textures on other instruments, i.e. extend our previous studies on other percussion instruments, piano, violin, and cello.
- Develop enhanced methods for visualization of both motion capture and EMG data, so as to better represent the multimodal nature for sound-motion textures.
- Design models and studies for intermittency in production control, production effort, and even perception of musical sound.

Acknowledgments

Many thanks to the participating musicians and to other staff for helping in the recording and processing of the performance data. Also, many thanks to two anonymous reviewers for very useful comments.

7. REFERENCES

[1] R. I. Godøy & M. Leman (Eds.), *Musical gestures: Sound, movement, and meaning*. Routledge, 2010.

[2] E. A. Franz, H. N. Zelaznik, S. Swinnen, & C. Walter, "Spatial conceptual influences on the coordination of bimanual actions: when a dual task becomes a single task," in *Journal of Motor Behavior 33*(1), 2001, pp. 103-112.

[3] S. T. Klapp, J. M. Nelson, & R. J. Jagacinski, "Can people tap concurrent bimanual rhythms independently?" in *Journal of Motor Behavior, 30*(4), 1998, pp. 301–322.

[4] M.-H. Song, & R. I. Godøy, "How Fast Is Your Body Motion? Determining a Sufficient Frame Rate for an Optical Motion Tracking System Using Passive Markers," in *PLoS ONE*, *11*(3), 2016, e0150993. E. A. http://doi.org/10.1371/journal.pone.0150993

[5] W. Goebl & C. Palmer, "Tactile feedback and timing accuracy in piano performance," in *Exp Brain Res*, *186*, 2008, 471–479.

[6] B. Burger & P. Toiviainen, "Mocap toolbox – a Matlab toolbox for computational analysis of movement

data," in *Proceedings of the sound and music computing conference, Stockholm. KTH Royal Institute of Technology*, 2013.

[7] M. Roerdink, A. Ridderikhoff, C. E. Peper & P. J. Beek, "Informational and Neuromuscular Contributions to Anchoring in Rhythmic Wrist Cycling," in *Annals of Biomedical Engineering*, *41*(8), 2013, pp. 1726–1739.

[8] R. I. Godøy, M.-H. Song, K. Nymoen, M. H. Romarheim, & A. R. Jensenius, "Exploring Sound-Motion Similarity in Musical Experience," in *Journal of New Music Research*, 45(3), 2016, pp. 210-222.

[9] R. I. Godøy, "Understanding coarticulation in musical experience," in M. Aramaki, M. Derrien, R. Kronland-Martinet, & S. Ystad (Eds.), *Sound, music, and motion. Lecture Notes in Computer Science*. Springer, 2014, pp. 535–547.

[10] S. T. Grafton & A. F. de C. Hamilton, "Evidence for a distributed hierarchy of action representation in the brain," in *Human Movement Science*, *26*(4), 2007, pp. 590–616.

[11] D. A. Rosenbaum, *Human Motor Control* (second edition). Academic Press Inc., San Diego, 2009).

[12] A. Karniel, "The minimum transition hypothesis for intermittent hierarchical motor control," *Frontiers in Computational Neuroscience*, *7*, 12. 2013. http://doi.org/10.3389/fncom.2013.00012

[13] S. Dahl, "Movements, timing and precision of drummers," in B. Müller, S. Wolf, G.-P. Brueggemann, Z. Deng, A. McIntosh, F. Miller, & W. S. Selbie (Eds.), *Handbook of Human Motion*, ISBN 978-3-319-30808-1 Springer (Forthcoming).

[14] S. T. Klapp & R. J. Jagacinski, "Gestalt principles in the control of motor action." in *Psychological Bulletin, 137*(3), 2011, pp. 443–462.

[15] I. D. Loram, van C. De Kamp, M. Lakie, H. Gollee, & P. J. Gawthrop, "Does the motor system need intermittent control?" in *Exercise and Sport Science Review, 42*(3), 2014, pp. 117–125.

Jazz Piano Trio Synthesizing System Based on HMM and DNN

Takeshi Hori
Meiji University
cs51003@meiji.ac.jp

Kazuyuki Nakamura
Meiji University
knaka@meiji.ac.jp

Shigeki Sagayama
Meiji Univerisity
sagayama@meiji.ac.jp

ABSTRACT

In this paper, we discuss a computational model of an automatic jazz session, which is a statistically trainable. Moreover, we describe a jazz piano trio synthesizing system that was developed to validate our model.

Most previous mathematical models of jazz session systems require heuristic rules and human labeling of training data to estimate the musical intention of human players in order to generate accompaniment performances. In contrast, our goal is to statistically learn the relationship between a piano, a bass, and a drum player from performance MIDI data as well as information contained in lead sheets, for instance tonic key and chord progression. Our system can generate the performance data of bass and drums from only piano MIDI input, by learning the interrelationship of their performances and time series characteristics of the three involved instruments. The experimental results show that the proposed system can learn the relationship between the instruments and generate jazz piano trio MIDI output from only piano input.

1. INTRODUCTION

We previously developed an automatic accompaniment system called Eurydice [1, 2], that can handle tempo changes and note insertion/deviation/substitution mistakes in the accompanied human performance as well as repeats and skips. Although this system can deal with various mistakes, the musical score of the accompaniment is fixed in Eurydice, since the system follows the player's performance by matching between the performance and the music score.

To appropriately accompany improvised performances, the accompaniment system needs to estimate the musical intention of the improvising player. Such systems are called jazz session systems. Previous jazz session systems [3–7] require heuristic rules and human labeling of training data to estimate the musical intention of human players. In contrast, our approach is to statistically learn the relationship between performances of the different instruments from performance MIDI data as well as information contained in lead sheets, in order to avoid the need of parameters set by the developer.

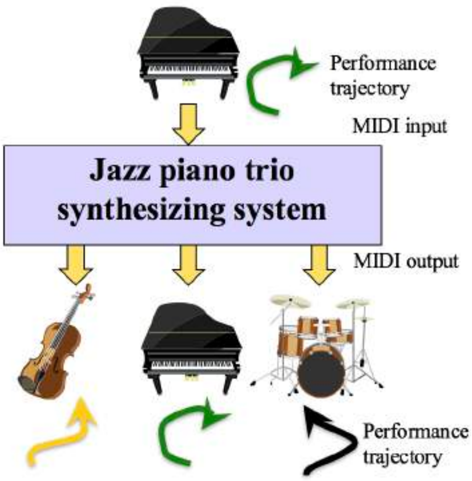


Figure 1. The input and output of the system described in this paper. The instrument performances are modeled as trajectories in a musical performance feature space, as described in section 2.1.

We previously developed an offline jazz piano trio synthesizing system [8] utilizing a database of performances from which appropriate accompaniment performances of bass and drums are selected based on characteristics of the piano MIDI input. Experimental results demonstrated the system's ability to meaningfully accompany piano performances to a certain degree.

This paper describes an offline jazz piano trio synthesizing system based on free generation of accompaniment performances without using the performance database. Instead, the system can learn the interrelationship of the instrument performances and their time series characteristics using hidden Markov models and deep neural networks. The approach follows a recent trend to uses deep learning (especially long short-term memory recurrent neural networks (LSTM-RNN [9]) in automatic music generation. The composition system DeepBach [10] uses a bidirectional LSTM to generate Bach-style music. Chu et al. [11] use a LSTM to generate POP music.

As illustrated in Fig. 1, the modeled jazz piano trio consists of a piano, whose performance MIDI data is used as input and a bass as well as drums whose performance MIDI data is generated by the computer.

2. MATHEMATICAL MODEL OF THE JAZZ SESSION SYSTEM

2.1 Trajectory model

We describe a musical performance as a trajectory in musical feature space [12]. At every point of time in the performance, the musical features of the current performance state (tempo, harmony, etc.) identify a position in the musical feature space. Consequently, a jazz musician "moves" through this feature space while performing, resulting in a performance trajectory.

In case of a jazz session comprising multiple instruments, one can observe a correlation between the trajectories of the different instruments during a "good session". Therefore, our aim is to implement a statistically trainable model that is able to learn the correlation between the instruments' trajectories. This correlation is used to generate the performance trajectories of accompaniment instruments (bass and drums) from the performance trajectory of the main instrument (piano).

2.2 The problems and solution strategies

Since the amount of available jazz performance data was relatively small, three problems had to be addressed in order to use machine learning methods effectively. Firstly, the number of musical features that could possibly be used to describe performances is large, resulting in a very high-dimensional feature space. To keep the dimensionality relatively low, we chose a set of features especially suited for jazz music. Secondly, since the data in feature space is sparse, an interpolation method is required to generate continuous performance trajectories through this space. This problem is addressed by segmenting the feature space into subspaces using a continuous mixture hidden Markov model. Thirdly, the jazz session system needs to learn the non-linear dependency between the performance trajectories of the different instruments, which is achieved using neural networks.

2.2.1 High dimensionality

We selected 68 features that are extractable from the music performance at every bar. The features were chosen based on opinions of jazz musicians, and are the following:

- *Piano-specific features*
  - The range between the highest and lowest notes.
  - The number of notes contained in the current diatonic scales, as well as the numbers of tension notes, avoid notes, and blue notes.
  - The ratios of diatonic notes, tension notes, avoid notes and blues notes to the total number of notes in the current bar.
- *Bass-specific features*
  - The range between the highest and lowest notes.
- *Drums-specific features*
  - The number of notes played by the hi-hat, snare drum, crash cymbal and the total number of all other notes.
- *Common features*
  - The number of notes, the number of simultaneously played notes, and the average MIDI velocities (loudness).
  - The ratio of off-beat notes to all notes in the current bar.
  - The ratios of all features (except for the off-beat-ratio) with respect to the previous bar as well as the ratios with respect to the complete performance.

We refer to these features as "style features" in the following.

2.2.2 Trajectory continuity

To generate performance trajectories from anywhere in feature space, one has to discretize or interpolate the available music performance data. In the proposed jazz session system, this is achieved by using clustering in the musical feature spaces of the accompaniment instruments. As a first step, the musical feature vectors are clustered using a continuous mixture hidden Markov model. The observation probabilities related to each hidden state of the HMM are described utilizing a Gaussian mixture model (GMM). Based on preliminary analysis of the data, the number of hidden states was chosen to be 4, using 4 GMM clusters per state. As a result, the trajectory model is approximated by a stochastic state transition model. A musical performance corresponds to a Viterbi path in the HMM and the interaction between instruments to co-occurrence of state transitions.

To increase precision, we further segment each HMM state internally. We are using a GMM to cluster the vectors of deviation from the HMM state's centroid.

The GMMs approximate a distribution as a mixture of normal distributions, and their updating algorithm applies the auxiliary function method. In case of the HMM, the state transition probabilities and the states' GMMs are optimized jointly using the auxiliary function method.

The auxiliary function has to fulfill Jensen's inequality, formulated as follows:

$$\begin{aligned} F(\sum_i \phi_i f_i(\boldsymbol{x};\theta)) &\geq \sum_i \phi_i F(f_i(\boldsymbol{x};\theta)) & (1)\\ &=: G(\theta,\phi) & (2)\\ s.t. \sum_i \phi_i &= 1, & (3) \end{aligned}$$

where $G(\theta,\phi)$ is the auxiliary function, $\phi = (\phi_1, \cdots, \phi_n)$ are the auxiliary variables, θ is a parameter, and $F(\cdot)$ a convex function. $\boldsymbol{x}$ is either a style feature vector (when obtaining the HMM states), a vector of deviation from an HMM state's centroid (for state internal clustering).

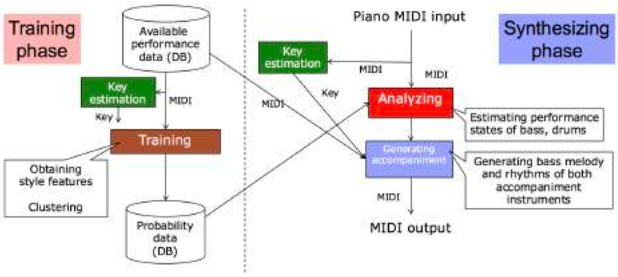


Figure 2. The training and synthesizing phases of the proposed system. During training, the instruments' interrelationship is learned and used to synthesize "improvised" accompaniment performances.

The functions $f_i(\cdot)$ encode the class assignments (HMM states or deviation clusters, respectively). In the auxiliary function method, the following two iterative update steps are applied alternately:

$$\phi^* = \arg\max_{\phi} G(\theta, \phi), \tag{4}$$

$$\theta^* = \arg\max_{\theta} G(\theta, \phi). \tag{5}$$

2.2.3 non-linear instrument performance interrelationship

A LSTM is used to learn the relationship between the musical features of the piano performance and the HMM segments of the bass and drums performance feature spaces. To learn the relationship between the piano features and vector of deviation from the states' centroids, a deep belief network (DBN) is used. The training process of these networks is described in the following section.

3. JAZZ PIANO TRIO SYNTHESIZING SYSTEM

Based on the stochastic state transition model which approximates a performance trajectory model, as described in the previous section, we developed an automatic jazz accompaniment system. The system receives MIDI-format data of only the piano performance, and as output generates MIDI-format data of the performance of the complete jazz piano trio (bass and drums added to the piano).

Fig. 2 illustrates the training and synthesizing phases of our system, which are described in the following.

3.1 Training phase

The training phase consists of following 6 steps:

1. *Key estimation*
   Estimating the tonic key of every bar of the performances in the training data.

2. *Feature extraction*
   Extracting performance features and computing the style features of all instruments at every bar.

3. *Data segmentation*
   Segmenting the performance feature spaces of bass and drums using a continuous mixture HMM.

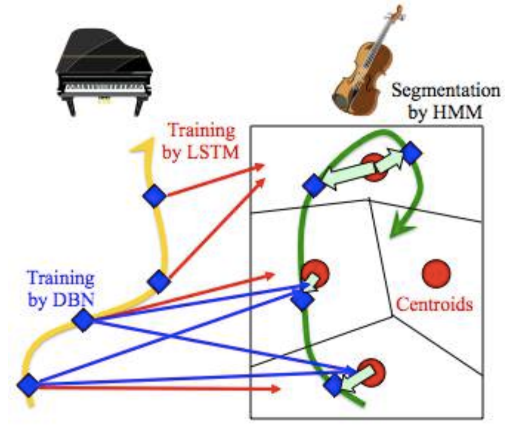


Figure 3. The training result of the proposed system. A LSTM matches piano performance vectors to the other instruments HMM states and a DBN refines the matching by providing deviation vectors from states' centroids.

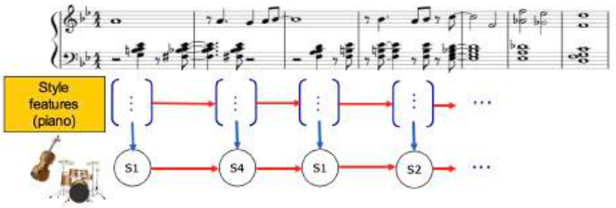


Figure 4. The LSTM matches style feature vectors of the piano performance to HMM states $(S_1, \cdots, S_4)$ of the other instruments.

4. *Performance correlation (coarse)*
   Training a LSTM network to match piano performance feature vectors to the other instruments' feature space segments derived in (3.).

5. *Clustering of centroid deviation vectors*
   Using a Gaussian mixture model (GMM) to cluster the vectors of deviation from the centroids of the HMM states derived in (3.).

6. *Performance correlation (fine)*
   Training a DBN [13] to match piano performance features vectors to the centroid deviation clusters derived in (5.).

Fig. 3 illustrates the training process described above.

3.1.1 Long short-term memory (LSTM)

A LSTM [9,14] is a type of recurrent neural network which can effectively handle time series data. In our system, a LSTM is used to match piano performance features to the HMM states which are obtained as clusters in other feature space of the two other instruments (see section 2.2.2). This essentially corresponds to identifying non-linear clusters in the piano performance feature space which are related to those of the other instruments. This dependency is illustrated as red arrows in Fig. 3. The LSTM is trained using MIDI data of jazz piano trio performances.

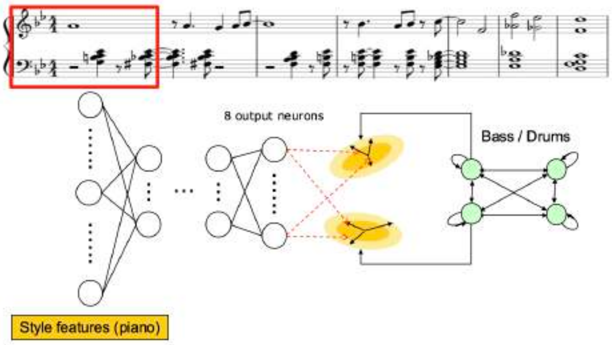


Figure 5. The DBN matches style feature vectors of the piano performance to deviation vectors from the centroids of the HMM states.

It is then used to estimate performance states of the accompanying instruments from the piano performance as shown in Fig. 4.

3.1.2 Deep belief network (DBN)

The used DBN consists of 5 restricted Boltzmann machine (RBM) layers and a classification output layer. Our jazz session system utilizes this DBN to match piano performance feature vectors to vectors of deviation from centroids of the HMM segments. As described in section 2.2.2, a GMM is applied to obtain 8 clusters of deviation vectors for each HMM state. The DBN effectively assigns a set of 4 deviation vectors (one for each HMM state) to every vector in the piano performance feature space (see Fig. 5).

We use Gaussian-Bernoulli RBM layers [15] in our system's DBN, because this type of RBM is able to handle continuous values, which our case are the style feature values extracted from the instrument performances. To update the RBM layers, we use the persistent contrastive divergence (PCD) method [16], because of its efficiency.

3.1.3 Combined results

The LSTM network and DBN in combination compute the tuple of HMM state and the associated centroid deviation vector for each piano performance vector. This tuple identifies a unique vector in the performance feature space of an accompaniment instrument, corresponding to a discretized step of the performance trajectory of this instrument. Two separate sets of networks are trained for the two accompaniment instruments (bass and drums).

3.2 Synthesizing phase: Generating bass and drums performances

In the synthesizing phase, we use the neural networks described in the previous section to estimate the performance states of bass and drums from a given piano performance feature vector. In the context of a complete performance, this corresponds to estimating the two accompaniment performance trajectories based on the piano performance trajectory.

To develop a method to automatically generate improvised performances from given performance trajectories, we consulted several jazz musicians. The devised method comprises the following three steps for the bass performance:

1. Constructing a melody without rhythm, based on the current harmony information, including the chord and information about usable notes (jazz scales, etc.).

2. Deciding on a rhythm pattern based on the current performance features. The rhythm is generated using a rhythm tree as explained in section 3.2.2.

3. Combining the melody with the rhythm.

In case of the drums, only the second step is required, where the rhythm is generated for every individual element (snare drum, hi-hat, etc.). The performance generation method is applied for every bar of the musical performance.

3.2.1 Generation of the bass melody

Since the melody of the bass generally doesn't include large interval jumps, we select melody notes close to the respective previous note. In detail, we compute the probability distribution of intervals occurring in the bass performances of the training data. This distribution is approximated as a Gaussian distribution with standard deviation $\sigma_{S,C}$ and obtained for every state S of the HMM and deviation vector cluster C. In the melody generation process, the probability of the ith MIDI note $x_{t,i}$ of bar number t is obtained as follows:

$$x_{t,i} \sim \mathcal{N}(\frac{1}{N_{t,i-1}} \sum_{j=0}^{i-1} x_{t,j}, (\frac{\sigma_{S,C}}{2})^2), \qquad (6)$$

where the average of all notes played in bar t before note $x_{t,i}$ is the mean of the Gaussian distribution. From this distribution, the notes are randomly sampled for every melody step.

3.2.2 Rhythm pattern generation

To generate the rhythms of the accompaniment instruments, we use idea of the rhythm tree [17]. In the rhythm tree, the root node corresponds to a whole note, which spans the whole bar. The tree then branches into nodes representing half notes or combinations of a half note and a half pause, and then further into quarter, eighth, and sixteenth note nodes. In an iteration of the rhythm pattern generation process, a note can be replaced by an associated child node. The number of iterations equals the number of tree branching layers (4 if branching until reaching sixteenth notes). A "rhythm characteristic" is a set of probability values that determine whether or with which child node a note is replaced in an iteration. The rhythm characteristic probabilities are extracted from the training data for every HMM state and deviation vector. In the synthesizing phase, the rhythm characteristics are then used to sample a rhythm pattern that fits to the performance state obtained as the tuple of HMM state and deviation vector from the neural networks.

4. EXPERIMENTAL EVALUATION

We trained our jazz piano trio synthesizing system using MIDI data of jazz piano trio performances of 12 different songs. We used the Python library Theano to implement the neural networks, whose properties are listed in table 1. The 12 training songs contain a total of about 2000 bars of which about 150 bars were used as validation set. The matching error of the LSTM, which matches piano performance vectors to performance states of the other instruments, was approximately 26% in case of the bass and 49% in case of the drums. The matching error of the DBN, which matches piano feature vectors to HMM state centroid deviation vector clusters, was approximately 28% in case of the bass and 38% in case of the drums.

Table 1. System Parameters

HMM parameters	
The number of state	4
The number of mixture	4
Epoch	100
GMM parameters (deviation from center)	
The number of mixture	8
Epoch	200
LSTM parameters	
Learning rate	0.0001
The number of middle layer units	30
Momentum	0.9
Drop out	0.5
Epoch	5000
DBN parameters	
Learning rate	0.00001
The number of hidden units (RBM)	50
Momentum	0.9
Weight decay	0.001
Objective average activation	0.01
Coefficient for sparse regularization	0.1
The number of layers	5
Prior epoch	10000
Epoch	100000

We let the system generate accompaniment performances for another song and asked 4 students to rate the result using a five-point grading scale (2 students had more than ten years of musical experience). We then compared the scores with evaluation results of previously used methods based on the selection of performance data from a database [18]. These methods include:

- Random: Only harmony constraints are met and both melody and rhythm pattern are drawn randomly from the performance database.

- Nearest neighbor: For each bar, the system searches the database to find the piano performance with the closest resemblance to the input piano performance, and the associated accompaniment performances are used for the respective bar.

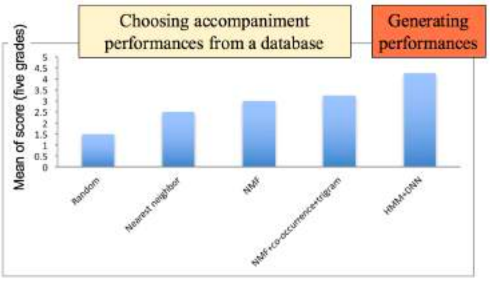


Figure 6. A result of comparative evaluation experimental. Every evaluation score is the average of the grades given by the four research participants.

- NMF: The piano performance feature space is clustered using non-negative matrix factorization (NMF) and the nearest neighbor method is applied within the resulting clusters.

- NMF + co-occurrence + trigram: The other instruments' feature spaces are clustered as well (using NMF). To identify the instrument performance interrelationship, co-occurrence matrices of the different instruments' performance feature clusters are obtained. Additionally, for each instrument, trigram probabilities are computed. The accompaniment performances are then chosen by maximizing the trigram probability combined with the co-occurrence probability.

The evaluation results are shown Fig. 6. A significant improvement over the previously used methods was observed. While the best previous method achieved a score of 3.25, the proposed system reached a score of 4.25.

5. CONCLUSION

We modeled a jazz piano trio session as the set of performance trajectories of the trio's instruments. To be able to learn from a small amount of data, we approximated the trajectory model by a stochastic state transition model. We used a LSTM network as well as a DBN to learn the correlation between the performance of the different jazz piano trio instruments from the performance MIDI data of jazz songs. Based on the estimated performance states of the accompaniment instruments, the system generates melody and rhythm of the accompaniment.

The proposed jazz session system received a higher evaluation score than our previous system. This confirms that the use of subspace clustering is an effective means to increase the precision of the used machine learning methods. However, there is still room for improvement, as the amount of available training data was relatively small and further investigation of the system and style features might be fruitful.

Another possibility to improve the system might be the use of context clustering. In our system, the context will be the musical structure or the harmony context (chord progression dependencies). Yoshimura et al. [19] have observed that context clustering can be useful to increase the effectiveness of machine learning methods utilized in natural speech synthesizing systems.

Furthermore, one could investigate how to obtain more training data, replace parametric clustering methods with non-parametric ones, or enable the system to handle audio input data instead of MIDI data. Moreover, our future goal is to develop a real-time jazz session system.

6. REFERENCES

[1] H. Takeda, N. T, S. Sagayama *et al.*, "Automatic acclonpaniment system of MIDI performance using HMM-based score followng," *IPSJ SIG Technical Reports*, pp. 109–116, 2006.

[2] E. Nakamura, H. Takeda, R. Yamamot, Y. Saito, S. Sako, S. Sagayama *et al.*, "Score following handling performances with arbitrary repeats and skips and automatic accompaniment," *IPSJ Journal*, vol. 54, no. 4, pp. 1338–1349, 2013.

[3] S. Wake, H. Kato, N. Saiwaki, and S. Inokuchi, "Cooperative musical partner system using tension - parameter : JASPER (jam session partner)," *IPSJ Journal*, vol. 35, no. 7, pp. 1469–1481, 1994.

[4] I. Hidaka, M. Goto, and Y. Muraoka, "An automatic jazz accompaniment system reacting to solo," *IPSJ SIG Technical Reports*, vol. 1995, no. 19, pp. 7–12, 1995.

[5] M. Goto, I. Hidaka, H. Matsumoto, Y. Kuroda, and Y. Muraoka, "A jazz session system for interplay among all playersI. system overview and implementation on distributed computing environment," *IPSJ SIG Technical Reports*, vol. 1996, no. 19, pp. 21–28, 1996.

[6] I. Hidaka, M. Goto, and Y. Muraoka, "A jazz session system for interplay among all playersII. implementation of a bassist and a drummer," *IPSJ SIG Technical Reports*, vol. 1996, no. 19, pp. 29–36, 1996.

[7] M. Hamanaka, M. Goto, H. Aso, and N. Otsu, "Guitarist simulator: A jam session system statistically learning player's reactions," *IPSJ Journal*, vol. 45, no. 3, pp. 698–709, 2004.

[8] T. Hori, K. Nakamura, and S. Sagayama, "Selection and concatenation of case data for piano trio considering continuity of bass and drums," *Autumn Meeting Acoustical Society of Japan*, vol. 2016, pp. 701–702, 2016.

[9] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[10] G. Hadjeres and F. Pachet, "Deepbach: a steerable model for bach chorales generation," *arXiv preprint arXiv:1612.01010*, 2016.

[11] H. Chu, R. Urtasun, and S. Fidler, "Song from pi: A musically plausible network for pop music generation," *arXiv preprint arXiv:1611.03477*, 2016.

[12] T. Hori, K. Nakamura, and S. Sagayama, "Statistically trainable model of jazz session: Computational model, music rendering features and case data utilization," *IPSJ SIG Technical Reports*, vol. 2016, no. 18, pp. 1–6, 2016.

[13] G. E. Hinton, S. Osindero, and Y. W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[14] A. Graves, "Supervised sequence labelling," in *Supervised Sequence Labelling with Recurrent Neural Networks*. Springer, 2012, pp. 5–13.

[15] Y. Freund and D. Haussler, "Unsupervised learning of distributions on binary vectors using two layer networks," in *Advances in neural information processing systems*, 1992, pp. 912–919.

[16] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[17] M. Tsuchiya, K. Ochiai, H. Kameoka, and S. Sagayama, "Probabilistic model of two-dimensional rhythm tree structure representation for automatic transcription of polyphonic midi signals," in *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2013 Asia-Pacific*. IEEE, 2013, pp. 1–6.

[18] T. Hori, K. Nakamura, and S. Sagayama, "Automatic selection and concatenation system for jazz piano trio using case data," *Proc. ISCIE International Symposium on Stochastic Systems Theory and its Applications*, vol. 2017, 2017.

[19] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura, "Simultaneous modeling of spectrum , pitch and duration in hmm-based speech synthesis," *The IEICE Transactions*, vol. 83, no. 11, pp. 2099–2107, 2000.

AUDIO-VISUAL SOURCE ASSOCIATION FOR STRING ENSEMBLES THROUGH MULTI-MODAL VIBRATO ANALYSIS

Bochen Li
University of Rochester
bli23@ur.rochester.edu

Chenliang Xu
University of Rochester
chenliang.xu@rochester.edu

Zhiyao Duan
University of Rochester
zhiyao.duan@rochester.edu

ABSTRACT

With the proliferation of video content of musical performances, audio-visual analysis becomes an emerging topic in music information retrieval. Associating the audio and visual aspects of the same source, or audio-visual source association, is a fundamental problem for audio-visual analysis of polyphonic musical performances. In this paper, we propose an approach to solve this problem for string ensemble performances by analyzing the vibrato patterns. On the audio side, we extract the pitch trajectories of vibrato notes of each string player in a score-informed fashion. On the video side, we track the left hand of string players and capture their fine-grained left-hand vibration due to vibrato. We find a high correlation between the pitch fluctuation and the hand vibration for vibrato notes, and use this correlation to associate the audio and visual aspects of the same players. This work is a complementary extension to our previous work on source association for string ensembles based on bowing motion analysis. Experiments on 19 pieces of chamber musical performances with at most one non-string instrument show more accurate and robust association performance than our previous method.

1. INTRODUCTION

Music performance, by its nature, is a multi-modal art form. Both the audio and visual aspects of music performance play an important role in engaging the audience in live concerts. Indeed, watching musical performances is often more entertaining than purely listening, since the players' motion visually interprets the melody and attracts the audiences. No wonder, for recorded music, the popularization of video streaming services such as YouTube, has also greatly shifted the dominating recording format from purely audio to video [1]. Despite the rich multi-modality of music, current Music Information Retrieval (MIR) research still mainly focuses on the audio modality of music, ignoring the visual aspects.

Joint analysis of audio and visual aspects of musical performances can significantly boost the state of the art of MIR research. Challenging tasks such as instrument recognition and activity detection in musical performances can be simplified by analyzing the visual aspects [2]. Joint analysis can also augment existing audio-based approaches to chord recognition [3] and onset detection [4] by tracking performers' hands and fingers . These benefits are especially prominent for analyses of polyphonic music, because the visual activity of each player is usually directly observable, whereas the polyphony makes it difficult to unambiguously associate components of the audio to each player [5]. Audio-visual analysis also opens up new frontiers for several existing and new MIR problems such as the analysis of sound articulation and performance expressiveness [6], fingering investigation [7], conductor following [8], source localization [9], etc.

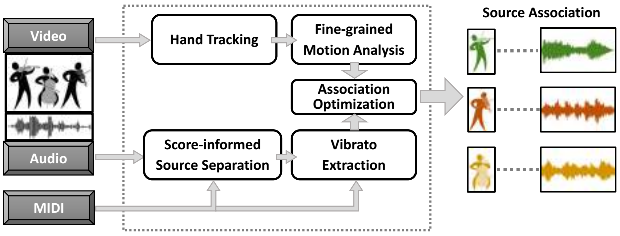


Figure 1. Framework of the proposed system. Separated sources are associated with players in the video, encoded with different colors.

A basic problem in audio-visual analysis of polyphonic music performances is to figure out the association between the audio and visual aspects of each sound source. This task was named *audio-visual source association* in our prior work [10]. It is an important problem because it is required if one wants to incorporate visual analysis in parsing the musical scene. On the application side, such association links audio editing (e.g., remixing) and video editing (e.g., recomposing) of sound sources. It enables cameras to automatically follow and take close-up shots of the instrumental part that is playing the main melody. It can also enable novel multimedia interactions such as music video streaming services that allow users to click on a player in the video and isolate/enhance the corresponding audio components.

In this paper, we propose an approach to solve the audio-visual source association problem for string ensemble performances through multi-modal analysis of vibrato notes. Specifically, we show that the fine-grained motion of the fingering hand has a strong correlation with the pitch fluctuation of vibrato notes, and we use this correlation to solve source association for string ensembles in a score-informed fashion. Fig. 1 shows the framework of the proposed approach. On the audio side, we first estimate the pitch trajectory of each note of each instrument from the audio mixture in a score-informed fashion. Then, we detect vibrato notes by inspecting the periodicity of the pitch fluctuation. On the visual side, we first track the left hand of each string player, and then capture the fine motion of the hand using optical flow, resulting in a 1-D hand displacement curve along the principal motion direction. For each source-player pair, we define a matching score based on the cross correlation between the concatenated pitch trajectories of all vibrato notes and the concatenated displacement curve segments in the corresponding time ranges. Finally, the bijection between sources and players that maximizes the overall matching score is output as the association result.

This is a complementary extension to our previous work on source association [11] that uses the correlation between bowing strokes and score note onsets. When vibrato is used, the proposed approach offers a more accurate and robust solution, as the correlation between the audio and visual aspects is now performed through the entire process of vibrato notes instead of only the onsets as in our previous work. This advantage is shown by the better association performance on 19 video recordings of chamber musical performances that use at most one non-string instruments. In the following, we first describe related work on audio-visual association in Section 2. We then describe the proposed approach in Section 3. We evaluate the system in Section 4 and conclude the paper in Section 5.

2. RELATED WORK

There has been some previous work on audio-visual source association for speech and music signals. Hershey and Movellan [9] first explored the audio-visual synchrony to locate speakers in the video. In [12], a region tracking algorithm was applied to segment the video into temporal evolution of semantic objects to be correlated with audio energy. Casanovas *et al.* [13] applied non-linear diffusion to extract the pixels whose motion is most consistent with changes of audio energy. In [14], source localization results are incorporated into graph-cut based image segmentation to extract the face region of the speaker. Other methods include time delayed neural networks [15], probabilistic multi-modal generative models [16], and canonical correlation analysis (CCA) [17, 18]. All of the abovementioned systems, however, assume that there is at most one active sound source at a time.

When multiple sources are active simultaneously, cross-modal association can be utilized to isolate sounds that correspond to each of the localized visual features. Barzelay and Schechner [19, 20] detected drastic changes (i.e., onsets of events) in audio and video and then used their coincidence to associate audio-visual components belonging to the same source of harmonic sounds. Sigg *et al.* [21] reformulated CCA by incorporating non-negativity and sparsity constraints on the coefficients of the projection directions, to locate sound sources in movies and separate them by filtering. In [22], audio and video modalities were decomposed into relevant structures using redundant representations for source localization. Segments where only one source is active are used to learn a timbre model for the separation of the source. These methods, however, only deal with mixtures with at most two active sources. The difficulty of source association increases dramatically as the source number increases.

The audio-visual source association problem for chamber music ensembles is more challenging since several sound sources are active almost all the time. In [11], we proposed the first approach to audio-visual source association for string ensembles in a score-informed fashion where there are up to five simultaneously active sources (e.g., string quintets). This approach analyzes the large-scale motion from bowing strokes of string-instrument players and correlates it with note onsets in the score track. The assumptions are that most note onsets correspond to the beginning of bowing strokes, and that different instrumental parts show different rhythmic patterns. When these assumptions are invalid, for example, when multiple notes are played within a single bow stroke (i.e., legato bowing) or when different parts show a similar rhythmic pattern, this approach becomes less robust.

The analysis of the fingering hand motion due to vibrato articulations and their correlation to pitch fluctuations in this paper is a complementary extension of [11]. It extends the large-scale motion analysis of the bowing hand to the fine-grained motion analysis of the fingering hand. It also extends the audio-visual correlation from the discrete note onsets to the entire processes of vibrato notes. As vibrato articulations are common practices for string instrument players (at least for skilled players), and the vibrato parameters (e.g., rate, depth, and phase) of different players are quite different, this vibrato cue provides robust and complementary information for audio-visual source association for string ensembles.

3. METHOD

3.1 Audio Analysis

The main goal of audio analysis is to detect and analyze vibrato notes in a score-informed fashion. There has been much work on vibrato analysis for both singing voices [23] and instruments [24], but most of them only deal with monophonic music. For methods that do deal with polyphonic music, they only analyze vibrato in the melody with the presence of background music [25]. The more challenging problem of vibrato detection and analysis for multiple simultaneous sources is, however, rarely explored. In this paper, we first separate sound sources using a score-informed source separation approach [26] and then perform pitch analysis on each source to extract and analyze vibrato notes.

3.1.1 Score-informed Source Separation

Musical sound sources are often strongly correlated in time and frequency, and without additional knowledge the separation of the sources is very difficult. When the musical score is available, as for many Western classical music pieces, the separation can be simplified by leveraging the score information. In recent years, many different models

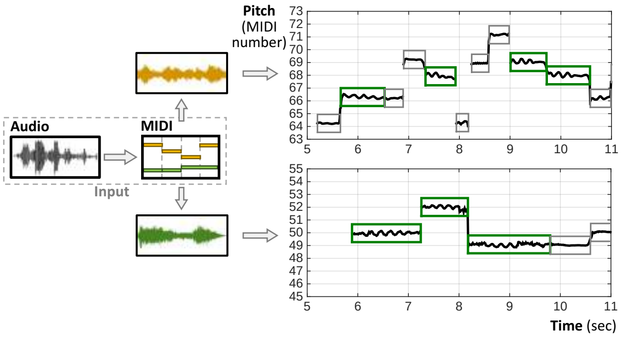


Figure 2. Process of audio analysis. Detected vibrato notes are marked with green rectangles in the pitch trajectories.

have been proposed to explore this idea, including Nonnegative Matrix Factorization (NMF) [27–29], Gaussian Mixture Models (GMM) [30], and adaptive instrument models [31]. In this paper, we adopt the pitch-based score-informed source separation approach of [26] and modify it for our task.

This approach consists of three main steps: audio-score alignment, pitch refinement, and source separation. The task of audio-score alignment is to guarantee the temporal synchronization between score events and audio articulations. We apply the Dynamic Time Warping (DTW) framework described in [11] instead of the online particle filtering approach in [26] for more accurate and robust alignment results.

The pitch refinement and source separation steps remain the same as those described in [26]: Pitches that are actually played in the music are first estimated within two semitones around the quantized score-notated pitches. Sound sources are then separated by harmonic masking of the pitches in each frame, where the soft masks take into account the harmonic indexes when distributing the mixture signals' energy to overlapping harmonics.

3.1.2 Vibrato Extraction

For each separated source, we apply the above-mentioned score-informed pitch refinement step again to estimate a robust pitch trajectory. We then segment the pitch trajectory into notes using the onset and offset information from the score. Vibrato notes are detected by checking the periodicity of their pitch trajectories. To do so, we borrow the idea of YIN pitch detection [32] to calculate the difference function of the pitch trajectory for each note i:

$$d_i(\tau) = \sum_{t=t_i^{on}}^{t_i^{off}-\tau} (F(t) - F(t+\tau))^2, \qquad (1)$$

where $F(t)$ is the estimated pitch trajectory of the source, t_i^{on} and t_i^{off} are the onset and offset frame indexes of the i-th note, and $\tau \in \{0, 1, \cdots, t_i^{off}\}$ is the time lag in the calculation. Then we calculate the normalized difference function $d'_i(\tau)$ using its cumulative mean as:

$$d'_i(\tau) = \begin{cases} 1 & \text{if } \tau = 0 \\ d_i(\tau)/[(1/\tau)\sum_{k=1}^{\tau} d(k)] & \text{otherwise} \end{cases} . \qquad (2)$$

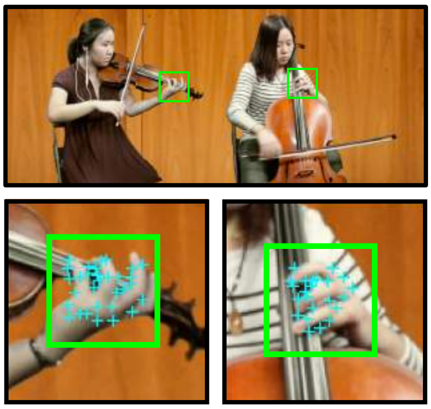


Figure 3. Hand tracking results. Feature points are marked by blue crosses, based on which the fingering hand is localized and bounded by a green bounding box.

As explained in [32], $\min_\tau d'_i(\tau)$ is a good indicator of whether periodicity exists or not, considering the energy of the signal. We use the threshold of 0.5, roughly tuned on one piece. If $\min_\tau d'_i(\tau)$ is smaller than the threshold, the note is considered to have vibrato. Fig. 2 highlights the pitch trajectories of the detected vibrato notes with green boxes.

It is noted that during the score-informed source separation process, the pitch trajectories of all sound sources have already been estimated. We do not, however, directly use this result, because we find that the pitch estimates are less accurate than those estimated from the separated sources. This suggests that the harmonic masking, although not perfect, does significantly reduce the interferences from other sources. It is also noted that the extracted vibrato pitch trajectories are still likely to contain errors, especially when multiple instruments play the same note. However, we will show that they are accurate enough for the audio-visual source association purposes.

3.2 Video Analysis

After retrieving the vibrato pitch trajectories from audio sources, we aim to extract the motion features from the visual aspect so that they can be correlated with the pitch fluctuations. A string instrument player mainly shows three kinds of motions when playing: bowing motion to articulate notes, fingering motion to control pitches, and the whole body motion to express musical intentions. Vibrato articulation is embodied as subtle motions on the left hand that rock forwards and backwards along the fingerboard. It changes the length and tension of the string. This rocking motions usually have a frequency ranging from 4 Hz to 7.5 Hz [33]. To capture this subtle motion, we first identify a region that covers the player's left hand, then estimate the rough position of the hand over time with a feature point-based tracker. Finally, we estimate the fine motion of the hand using optical flow analysis within the hand region.

3.2.1 Hand Tracking

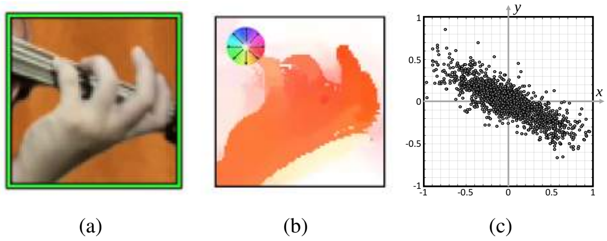
Given the bounding box initialization of the hand position, we apply the Kanade-Lucas-Tomasi (KLT) tracker to track the rough position of the left hand. This method was originally proposed in [34]. It extracts feature points using the minimum eigenvalue algorithm in a given region specified in the first video frame and then track these points throughout the video based on assumptions such as brightness constancy and spatial coherence. For our implementation (see Fig. 3), we manually specify a rectangle bounding box (70 × 70 pixels) to cover the left hand in the first frame and initialize 30 feature points within the bounding box using corner-point detector. We then track the points in following frames and relocate the center of the bounding box to the median x- and y- coordinates of the points. In order to prevent the loss of tracking due to abrupt motion or occlusions, we re-initialize the feature points every 20 frames, if some points disappear.

3.2.2 *Fine-grained Motion Capture*

To capture the fine motion of the left hand, we apply optical flow estimation [35] on the original video frames to calculate pixel-level motion velocities. For each frame, the average motion velocity over all pixels within the bounding box is calculated as $\mathbf{u}(t) = [u_x(t), u_y(t)]$. Notice that the pixel motions in the bounding box contain not only the player's hand vibrato motion but also his/her overall slower body motion while performing the music. This motion not only contains the hand vibration due to vibrato, but it also contains the overall slower body movement that is not associated with vibrato. In order to eliminate the body movement and obtain a clean description of the vibrato motion, we subtract $\mathbf{u}(t)$ by a moving average of itself, as

$$\mathbf{v}(t) = \mathbf{u}(t) - \bar{\mathbf{u}}(t), \tag{3}$$

where $\bar{\mathbf{u}}(t)$ is the moving average of $\mathbf{u}(t)$ by averaging over 10 frames. Fig. 4 displays one example with the original hand region in Fig. 4(a) and the optical flow result in Fig. 4(b). The refined motion velocity $\mathbf{v}(t)$ across all the frames is scattered in an x-y plane in Fig. 4(c).



(a) (b) (c)

Figure 4. Fine-grained left hand motion detection from optical flow. (a) Original video frame showing the hand region. (b) Optical flow results encoding motion velocities via colors. (c) Scatter plot of frame-wise average motion velocities of all frames.

We apply Principal Component Analysis (PCA) on $\mathbf{v}(t)$ across all frames to identify the principal direction of motion, which is ideally along the fingerboard. The refined motion velocity vectors are then projected to this principal direction to obtain a 1-D *motion velocity curve* $V(t)$ as

$$V(t) = \frac{\mathbf{v}(t)^T\tilde{\mathbf{v}}}{\|\tilde{\mathbf{v}}\|}, \tag{4}$$

where $\tilde{\mathbf{v}}$ is the eigenvector corresponding to the largest eigenvalue of the PCA of $\mathbf{v}(t)$. We then perform an integration of the motion velocity curve over time to calculate a *motion displacement curve* as

$$X(t) = \int_0^t V(\tau)d\tau. \tag{5}$$

This hand displacement curve corresponds to the fluctuation of the vibrating length of the string and hence the pitch fluctuation of the note. Fig. 5 plots the extracted motion displacement curve $X(t)$ of one player along with the pitch trajectory. We find that when vibrato is detected from audio (labeled in green rectangles), there are always observable fluctuations in the motion.

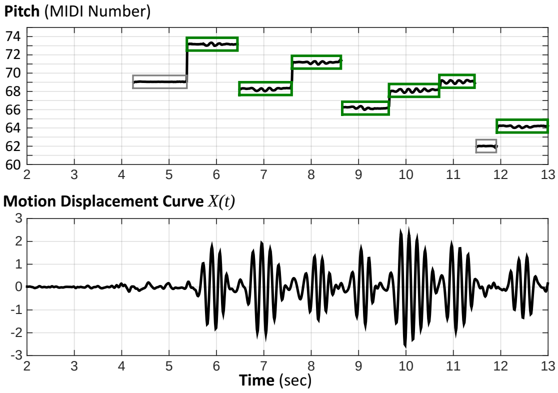


Figure 5. Pitch trajectory with detected vibrato notes marked by green rectangles from audio analysis and the left-hand motion curve of the corresponding player from video analysis.

3.3 Source-Player Association

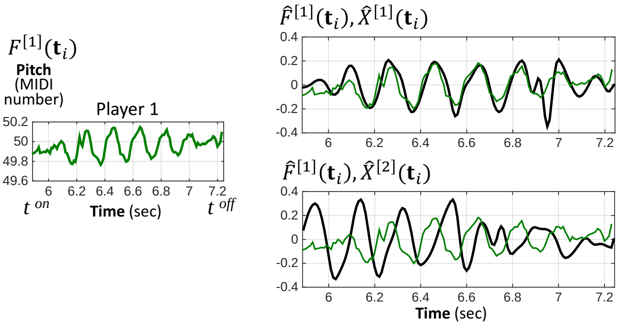


Figure 6. Pitch trajectory of one vibrato note (left), and the normalized left-hand displacement curves of the two players (right, black curves) overlaid with the normalized pitch trajectory (green curves). Note how well the pitch trajectory matches the displacement curve of Player 1 (correct association) but not of Player 2 (incorrect association).

The audio-visual association is a bijection between players and audio sources. This is obtained by matching the pitch trajectories and the visual displacement curves at the note level. Fig. 6 compares the pitch trajectory of one vibrato note with the displacement curves of two players at the same time range. All of the curves have been normalized to have zero mean and unit Root-Mean-Square (RMS) value. Clearly, Player 1 is a better match. Quantitatively, we measure the similarity between the pitch trajectory of each note of the p-th source and the hand displacement curve of the q-th player in the corresponding time range by

$$m^{[p,q]}(i) = \exp\left\{ \frac{\left\langle \hat{F}^{[p]}(\mathbf{t}_i) \cdot \hat{X}^{[q]}(\mathbf{t}_i) \right\rangle}{\left\| \hat{F}^{[p]}(\mathbf{t}_i) \right\| \left\| \hat{X}^{[q]}(\mathbf{t}_i) \right\|} \right\}, \tag{6}$$

where $\mathbf{t}_i = [t_i^{on}, t_i^{off}]$ represents the time range of the i-th note from its onset t_i^{on} to offset t_i^{off}, $\hat{F}$ and $\hat{X}$ are the normalized pitch trajectories and hand displacement curves, respectively, and $\langle \cdot \rangle$ represents the vector inner product. Then the overall matching score between the p-th source and the q-th player can be defined as the sum of the similarities of all vibrato notes:

$$M^{[p,q]} = \sum_{i=1}^{N_{\text{vib}}^{[p]}} m^{[p,q]}(i), \tag{7}$$

where $N_{\text{vib}}^{[p]}$ is the number of detected vibrato notes of the p-th source.

For an ensemble with K players, there are $K!$ bijective associations (permutations). Let $\sigma(\cdot)$ be a permutation function, then the p-th audio source will be associated with the $\sigma(p)$-th player. For each permutation, we calculate an overall *association score* as the product of all pair-wise matching scores, i.e., $S_\sigma = \prod_{p=1}^{K} M^{[p,\sigma(p)]}$. We then rank all of the association candidates in a descending order and return the first candidate as the final association.

4. EXPERIMENTS

We evaluate the proposed approach on the University of Rochester Musical Performance (URMP) dataset [1] [36]. This dataset contains isolately recorded but synchronized audio-visual recordings of all instrumental parts of 44 classical ensemble pieces ranging from duets to quintets. For our experiments, we use the 11 string ensemble pieces and 8 other pieces that use only one non-string instrument, totaling 19 pieces including 5 duets, 4 trios, 7 quartets, and 3 quintets. Fig. 8 shows screen shots of the performance videos of all the 19 test pieces. Most pieces have a length of 2-3 minutes, providing enough cues for source association. Detailed information about these pieces is listed in Table 1.

Audio is sampled at 48 KHz, and processed with a frame length of 42.7 ms and a hop size of 10 ms for the Short-Time Fourier Transform (STFT). Video resolution is 1080P, and the frame rate is 29.97 frames per second. The audio and video streams have been carefully synchronized in the dataset. The estimated pitch trajectories are interpolated to make the first pitch estimate correspond to time at 0 ms. The estimated hand displacement curves are also interpolated to make them have the same sample rate as the pitch trajectories, i.e., 1 sample every 10 ms. For non-string instruments where the vibrato is not generated by hand motion, we do not track the hand nor capture the motion; Instead, we set the hand displacement curve of these players in Eq. (5) to all zeros.

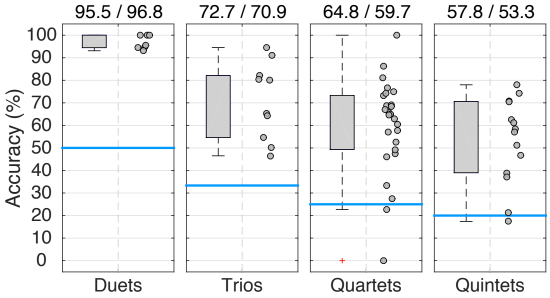


Figure 7. Box plots and scatter plots of note-level matching accuracy of string-instrumental sources of all pieces grouped by polyphony. Each data point represents the accuracy of one source. The blue lines mark the random guess accuracies. Numbers above the figures are median/mean values.

4.1 Note-level Matching Accuracy

As the matching score matrix $M^{[p,q]}$ used to find the best source association is the sum of the matching scores of all vibrato notes $m^{[p,q]}(i)$, we first evaluate how good the note-level matching scores are. To do so, we calculate a *note-level matching accuracy*, which is defined as the percentage of vibrato notes of each source that best matches the correct player according to the note-level matching score. Fig. 7 shows the boxplots and scatter plots of this measure of all sources (excluding the non-string instruments) of all pieces, grouped by their polyphony. Each point represents the accuracy of one source, and the numbers above show the median/mean values. The blue lines mark the accuracies based on random guess, e.g., the chance to randomly match a note to the correct player in a quartet is 25%. We find that this accuracy drops as the polyphony increase because of 1) the more incorrect candidates and 2) the less good quality of the extracted pitch trajectory of vibrato notes from high-polyphonic audio mixtures. Outliers in the figures are sources that only contain a few vibrato notes.

It is noted that a less good note-level matching accuracy does not necessarily lead to a bad piece-level source association, because the latter considers all the vibrato notes in a piece. In fact, as long as the note-level matching is somewhat accurate, the ensemble decision considering all of the vibrato notes at the piece-level is quite reliable, as illustrated in the following experiment.

4.2 Piece-level Source Association Results

We then evaluate the piece-level source association accuracy. The estimated association for each piece is considered correct if and only if all the sources are associated to the correct players. As the number of permutations is the factorial of the polyphony, this task becomes much

[1] http://www.ece.rochester.edu/projects/air/projects/datasetproject.html

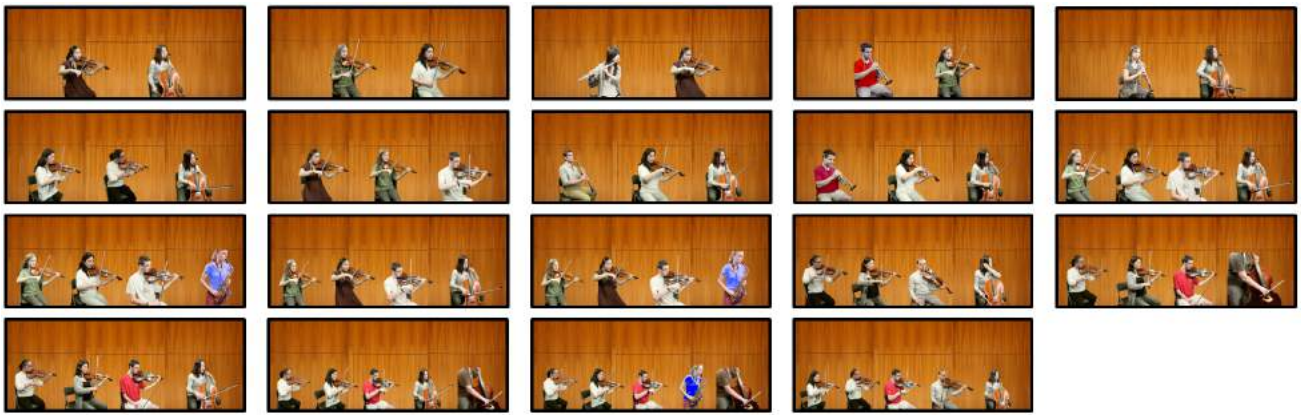


Figure 8. Screen shots of the 19 test pieces. Due to limited conditions of the recording facilities, all players faced to the same direction and some parts of certain players were not captured by the camera. These artifacts might have made the visual analysis of left-hand motion easier than in real-world performance recordings.

Metadata				**Association Measures**	
No.	Dataset Folder Name (with Instrument Types)	Piece Length (mm:ss)	Polyphony - (No. Permutations)	No. Correctly Associated Sources	Rank of Correct Association
1	01_Jupiter_vn_vc	01:03	2 - (2)	2	1
2	02_Sonata_vn_vn	00:46	2 - (2)	2	1
3	08_Spring_fl_vn	00:35	2 - (2)	2	1
4	09_Jesus_tpt_vn	03:19	2 - (2)	2	1
5	11_Maria_ob_vc	01:44	2 - (2)	2	1
6	12_Spring_vn_vn_vc	02:11	3 - (6)	3	1
7	13_Hark_vn_vn_va	00:47	3 - (6)	3	1
8	19_Pavane_cl_vn_vc	02:13	3 - (6)	1	2
9	20_Pavane_tpt_vn_vc	02:13	3 - (6)	3	1
10	24_Pirates_vn_vn_va_vc	00:50	4 - (24)	4	1
11	25_Pirates_vn_vn_va_sax	00:50	4 - (24)	4	1
12	26_King_vn_vn_va_vc	01:25	4 - (24)	4	1
13	27_King_vn_vn_va_sax	01:25	4 - (24)	2	1
14	32_Fugue_vn_vn_va_vc	02:54	4 - (24)	4	1
15	35_Rondeau_vn_vn_va_db	02:08	4 - (24)	4	1
16	36_Rondeau_vn_vn_va_vc	02:08	4 - (24)	4	1
17	38_Jerusalem_vn_vn_va_vc_db	01:59	5 - (120)	5	1
18	39_Jerusalem_vn_vn_va_sax_db	01:59	5 - (120)	5	1
19	44_K515_vn_vn_va_va_vc	03:45	5 - (120)	5	1

Table 1. Detailed information of the 19 test pieces together with the source association performance of the proposed approach. Abbreviations of instruments are: violin (Vn.), viola (Va.), cello (Vc.), double bass (D.B.), flute (Fl.), oboe (Ob.), clarinet (Cl.), saxphone (Sax.), and trumpet (Tp.).

more challenging for pieces with a high polyphony. In our experiments, the proposed approach successfully outputs the correct association for 18 of the 19 pieces, yielding a success rate of **94.7%**. This outperforms our previous method [11] whose accuracy was 89.5% on the same pieces. Among the 19 pieces, there are in total 65 individual sources, and 63 of them (96.9%) are associated with the correct player, while in our previous system only 58 sources are correct.

Table 1 lists the association results on all of the testing pieces. For each piece, we show the number of correctly associated sources and the rank of correct association among all permutations. The proposed approach only fails gently on one piece (No. 8), where the correct association is ranked the second. This failure case is a trio containing a clarinet while the two string instruments are always playing plucking notes, for which vibrato is not generally used.

The main reason that the proposed approach outperforms our previous system [11] on this dataset, as we stated in the introduction, is that the vibrato analysis provides more reliable cues for association than the note onset analysis in our previous system. Vibrato, if present, often occurs throughout a note; Vibrato patterns of different players are also distinct even if they are playing notes with the same rhythm. We have to admit that for performances that vibrato is not used, the proposed approach would fail. Nonetheless, it provides a complementary approach to our previous system and combining these two approaches is a future direction of research.

5. CONCLUSIONS

We proposed a methodology for source association by synergistic analyses of vibrato articulations from both audio and video modalities for string ensembles. Specifically, we found that the fine-grained motion of the fingering hand shows a strong correlation with the pitch fluctuation of vibrato notes, which was utilized to solve source association in a score-informed fashion. Experiments showed a high success rate on pieces of different polyphony, and proved that vibrato features provide more robust cues for source association than bowing motion features that were utilized in our previous work. This technique enables novel and richer music enjoyment experiences that allow users to isolate/enhance sound sources by selecting the players in the video. For future work, we would like to combine the vibrato cues and the bowing cues to achieve more robust association results. We also would like to explore scenarios where the score is not available.

6. REFERENCES

[1] C. Vernallis, *Unruly media: YouTube, music video, and the new digital cinema*. Oxford University Press, 2013.

[2] A. Bazzica, C. C. Liem, and A. Hanjalic, "Exploiting instrument-wise playing/non-playing labels for score synchronization of symphonic music." in *Proc. Intl. Society for Music Information Retrieval (ISMIR)*, 2014.

[3] A. Hrybyk, "Combined audio and video analysis for guitar chord identification," Master's thesis, Drexel University, 2010.

[4] A. Bazzica, J. C. van Gemert, C. C. Liem, and A. Hanjalic, "Vision-based detection of acoustic timed events: a case study on clarinet note onsets," in *International Workshop on Deep Learning for Music (DLM) in conjunction with the International Joint Conference on Neural Networks (IJCNN)*, Anchorage, Alaska (USA), 2017.

[5] K. Dinesh, B. Li, X. Liu, Z. Duan, and G. Sharma, "Visually informed multi-pitch analysis of string ensembles," in *Proc. IEEE Intl. Conf. Acoust., Speech, and Signal Process. (ICASSP)*, 2017.

[6] S. Dahl and A. Friberg, "Visual perception of expressiveness in musicians' body movements," *Music Perception: An Interdisciplinary Journal*, vol. 24, no. 5, pp. 433–454, 2007.

[7] J. Scarr and R. Green, "Retrieval of guitarist fingering information using computer vision," in *Proc. Intl. Conf. Image and Vision Computing New Zealand (IVCNZ)*, 2010.

[8] D. Murphy, "Tracking a conductor's baton," in *Proc. Danish Conf. Pattern Recognition and Image Analysis*, 2003.

[9] J. R. Hershey and J. R. Movellan, "Audio vision: Using audio-visual synchrony to locate sounds." in *Proc. Neural Inform. Process. Syst.*, 1999, pp. 813–819.

[10] B. Li, Z. Duan, and G. Sharma, "Associating players to sound sources in musical performance videos," in *Late Breaking Demo, Intl. Society for Music Information Retrieval (ISMIR)*, 2016.

[11] B. Li, K. Dinesh, Z. Duan, and G. Sharma, "See and listen: Score-informed association of sound tracks to players in chamber music performance videos," in *Proc. IEEE Intl. Conf. Acoust., Speech, and Signal Process. (ICASSP)*, 2017.

[12] K. Li, J. Ye, and K. A. Hua, "What's making that sound?" in *Proc. ACM Intl. Conf. Multimedia*, 2014, pp. 147–156.

[13] A. L. Casanovas and P. Vandergheynst, "Audio-based nonlinear video diffusion," in *Proc. IEEE Intl. Conf. Acoustics, Speech, and Signal Process. (ICASSP)*. IEEE, 2010, pp. 2486–2489.

[14] Y. Liu and Y. Sato, "Finding speaker face region by audiovisual correlation," in *Proc. Workshop on Multi-camera and Multi-modal Sensor Fusion Algorithms and Applications*, 2008.

[15] R. Cutler and L. Davis, "Look who's talking: Speaker detection using video and audio correlation," in *Proc. IEEE Intl. Conf. Multimedia and Expo,*, vol. 3, 2000, pp. 1589–1592.

[16] J. W. Fisher and T. Darrell, "Speaker association with signal-level audiovisual fusion," *IEEE Trans. Multimedia*, vol. 6, no. 3, pp. 406–413, 2004.

[17] E. Kidron, Y. Y. Schechner, and M. Elad, "Cross-modal localization via sparsity," *IEEE Trans. Signal Process.*, vol. 55, no. 4, pp. 1390–1404, 2007.

[18] H. Izadinia, I. Saleemi, and M. Shah, "Multimodal analysis for identification and segmentation of moving-sounding objects," *IEEE Trans. Multimedia*, vol. 15, no. 2, pp. 378–390, 2013.

[19] Z. Barzelay and Y. Y. Schechner, "Harmony in motion," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2007, pp. 1–8.

[20] Z. Barzelay and Y. Y. Schechner, "Onsets coincidence for cross-modal analysis," *IEEE Trans. Multimedia*, vol. 12, no. 2, pp. 108–120, 2010.

[21] C. Sigg, B. Fischer, B. Ommer, V. Roth, and J. Buhmann, "Nonnegative CCA for audiovisual source separation," in *Proc. IEEE Workshop on Machine Learning for Signal Processing*, 2007, pp. 253–258.

[22] A. L. Casanovas, G. Monaci, P. Vandergheynst, and R. Gribonval, "Blind audiovisual source separation based on sparse redundant representations," *IEEE Trans. Multimedia*, vol. 12, no. 5, pp. 358–371, 2010.

[23] L. Regnier and G. Peeters, "Singing voice detection in music tracks using direct voice vibrato detection," in *Proc. IEEE Intl. Conf. Acoust., Speech, and Signal Process. (ICASSP)*, 2009, pp. 1685–1688.

[24] S. Rossignol, P. Depalle, J. Soumagne, X. Rodet, and J.-L. Collette, "Vibrato: detection, estimation, extraction, modification," in *Proc. Digital Audio Effects Workshop (DAFx)*, 1999, pp. 1–4.

[25] J. Driedger, S. Balke, S. Ewert, and M. Müller, "Template-based vibrato analysis of music signals," in *Proc. Intl. Society for Music Information Retrieval (ISMIR)*, 2016.

[26] Z. Duan and B. Pardo, "Soundprism: An online system for score-informed source separation of music audio," *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 6, pp. 1205–1215, 2011.

[27] R. Hennequin, B. David, and R. Badeau, "Score informed audio source separation using a parametric model of non-negative spectrogram," in *Proc. IEEE Intl. Conf. Acoust., Speech, and Signal Process. (ICASSP)*, 2011.

[28] J. Fritsch and M. D. Plumbley, "Score informed audio source separation using constrained nonnegative matrix factorization and score synthesis," in *Proc. IEEE Intl. Conf. Acoust., Speech, and Signal Process. (ICASSP)*, 2013, pp. 888–891.

[29] S. Ewert and M. Müller, "Using score-informed constraints for NMF-based source separation," in *Proc. IEEE Intl. Conf. Acoust., Speech, and Signal Process. (ICASSP)*, 2012, pp. 129–132.

[30] P. Sprechmann, P. Cancela, and G. Sapiro, "Gaussian mixture models for score-informed instrument separation," in *Proc. IEEE Intl. Conf. Acoust., Speech, and Signal Process. (ICASSP)*, 2012, pp. 49–52.

[31] F. J. Rodriguez-Serrano, Z. Duan, P. Vera-Candeas, B. Pardo, and J. J. Carabias-Orti, "Online score-informed source separation with adaptive instrument models," *Journal of New Music Research*, vol. 44, no. 2, pp. 83–96, 2015.

[32] A. De Cheveigné and H. Kawahara, "Yin, a fundamental frequency estimator for speech and music," *The Journal of the Acoustical Society of America*, vol. 111, no. 4, pp. 1917–1930, 2002.

[33] J. M. Geringer, R. B. MacLeod, and M. L. Allen, "Perceived pitch of violin and cello vibrato tones among music majors," *Journal of Research in Music Education*, vol. 57, no. 4, pp. 351–363, 2010.

[34] C. Tomasi and T. Kanade, *Detection and tracking of point features*. School of Computer Science, Carnegie Mellon Univ. Pittsburgh, 1991.

[35] D. Sun, S. Roth, and M. J. Black, "Secrets of optical flow estimation and their principles," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2010.

[36] B. Li, X. Liu, K. Dinesh, Z. Duan, and G. Sharma, "Creating a musical performance dataset for multimodal music analysis: Challenges, insights, and applications," *IEEE Trans. Multimedia*, submitted. Available: https://arxiv.org/abs/1612.08727.

Investigations on stage acoustic preferences of solo trumpet players using virtual acoustics

Sebastià V. Amengual Garí
Detmold University of Music
amengual@hfm-detmold.de

Malte Kob
Detmold University of Music
kob@hfm-detmold.de

Tapio Lokki
Aalto University
tapio.lokki@aalto.fi

ABSTRACT

Acoustical conditions on stage have an influence on the sound and character of a musical performance. Musicians constantly adjust their playing to accommodate to the stage acoustics. The study of acoustical preferences of musicians is part of the characterization of this feedback loop, which impacts on the musicians' comfort as well as on the aural experience of a concert audience.

This paper presents an investigation on preferences of solo musicians on stage. By means of spatial acoustic measurements and real-time auralization, the acoustics of different real rooms are resynthesized in laboratory conditions. Two formal tests are conducted with solo trumpet players: a room preference test and a test investigating the preferred directions of early energy. In the first test, musicians are presented with four different rooms and asked about their preference in five different aspects: practice of instrument technique, practice of concert repertoire, concert performance, ease of performance and sound quality. In the second test the auralized rooms are modified to provide early reflections from different directions (front-back, top-down, sides, no early reflections) and the preference of musicians is investigated.

The results show that the judged aspect or performance context is a key factor in determining the preference of musicians' stage acoustics preference. Drier rooms are preferred for practicing instrumental technique while louder rooms help to reduce the fatigue of the players. Bigger rooms with slightly longer reverberation are preferred for concert piece practice and concert performance. The easiness of performance is proportional to the amount of early energy. Regarding the preference of direction of early reflections, the results suggest that there are no clear differences between preferred directions, and the level of early energy is more important for the comfort of solo musicians on stage.

1. INTRODUCTION

Stage acoustic preferences and their impact on musicians have been traditionally studied through two approaches: *in situ* experiments in real rooms or synthesis of room acoustics in laboratory conditions. Lately, the proliferation of room enhancement systems allowed as well the conduction of experiments in hybrid or semi-virtual conditions [1].

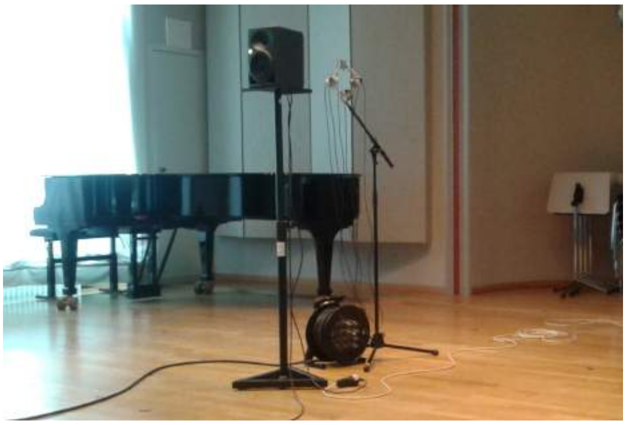


Figure 1: Spatial measurements in Brahmssaal (BS).

The main limitation of laboratory conditions in early experiments was to provide a plausible and realistic acoustic scene using electroacoustic systems. However, given those limitations and the challenge of implementing such virtual environments in real-time, Gade [2] was able to formulate a now widely accepted stage parameter, *stage support* (ST1) from a series of experiments in which a virtual room was generated by mixing the direct sound of a musician with its sound played in a real reverberation room. With the explosion of digital signal processing and increase of computing performance, successive approaches aimed at capturing and reproducing real rooms in laboratory conditions. Ueno *et al.* [3] used a 6-channels system based on directional impulse responses, while a system based on ambisonics reproduction was used by Guthrie *et al.* [4].

This paper uses the real-time auralization system introduced in [5] to conduct experiments on stage acoustics preferences of solo musicians. The auralization method is validated and a method to modify the directional response of an auralized room is proposed. Two experiments are described: a study of room acoustics preference depending on the performance context and a study on the preference of directional early energy.

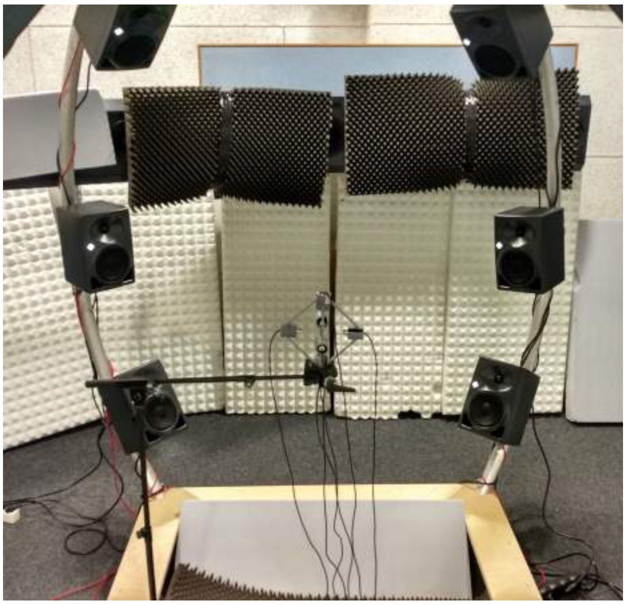


Figure 2: Validation measurement in the listening environment.

2. ROOM AURALIZATION

2.1 Measured rooms

Stage spatial measurements were conducted in three rooms of the Detmold University of Music, Germany:

- Brahmssaal (BS): small performance room (approx. 100 seats), regularly used for instrumental lessons and solo performances.
- Detmold Sommertheater (DST): Theater (approx. 320 seats), commonly used for theater, opera and ensemble performances.
- Detmold Konzerthaus (KH): Concert hall (approx. 600 seats) which hosts classical music concerts (ensembles and symphonic music), solo concert examinations and organ recitals.

A measurement set-up composed of a directional studio monitor (Neumann KH120 A) and a microphone array (6 omnidirectional measurement microphones – NTi M2010) is placed on stage at a relative distance of 62 cm, imitating a solo trumpet player situation (Fig. 1 shows the set-up in BS). Spatial Room Impulse Responses (SRIR) are derived and analyzed using the spatial decomposition method (SDM) [6]. An analyzed SRIR consists of an impulse response, in which every sample (pressure value) is associated with a direction (metadata). In auralization the samples of this single impulse response are distributed to all reproduction loudspeakers according to the direction metadata. As the spatial resolution of reproduction loudspeakers is sparse, final mapping of samples to reproduction loudspeakers is done with Vector Base Amplitude Panning [7] (VBAP). Finally, each reproduction loudspeaker has a sparse impulse response, which is convolved in real time with the input signal of a musician. The direct sound is removed from the convolution filters, as it is generated by the instrumentalists. The floor reflection is removed as well, in order to allow the convolution engine to use slightly larger buffer sizes, ensuring the stability of the system. The reproduction of the auralized scenes is achieved using a set-up composed of 13 reproduction loudspeakers in a quasi anechoic room with reverberation times less than 0.1 seconds at mid and high frequencies. Details on the measurements, auralization process, and reproduction set-up are described in [5] and are out of the scope of this paper.

2.2 Validation of the auralization

In order to validate the auralization system and compare the acoustic properties of the real and the auralized rooms, spatial measurements were conducted in the virtual environment. The convolution engine was fed with a swept sine signal and measured using the microphone array inside the listening environment (see Fig. 2). The same process as in the real rooms was followed to derive and analyze SRIR of the auralized rooms. Frequency response, auralization error and monaural room parameters (T20, C80) are presented in Fig. 3. Broadband spatiotemporal plots [8] (backward integrated) are presented in Fig. 4. As can be extracted from the measurements, the frequency response of the real and auralized rooms present an error smaller than ± 3 dB in the range of 200 Hz to 5 kHz, approximately. The monaural parameters present a good agreement in the same range, in most cases below the just noticeable difference threshold. The spatiotemporal plots represent the acoustic energy arriving at the player position in the real and auralized rooms, excluding the direct sound and the floor reflection. The analysis show a good agreement in spatial terms, except for room DST, whose auralization presents stronger early reflections from the back side. The increase of diffuse reverberation above 5 kHz can be compensated [9] and a correction routine is provided in the SDM Toolbox [10]. However, the correction procedure was not available when the presented experiments were completed and it will be considered for future experiments. In addition, the overall decrease of brightness is due to non-coherent summation of signals in amplitude panning at high frequencies [11].

2.3 Spatiotemporal manipulation

Once the directional information of an impulse response is computed the SRIR can be manipulated to modify the magnitude and directions of specific reflections. In this case, the early reflections of the room are manipulated to modify their energy and directions of arrival. To do that, the rendered impulse responses are split into an early reflections part (subject to spatial manipulation) and an unmodified late reverberation part. A directional weighting function is applied on the early reflections using a time window. The modified impulse response is a combination of modified early reflections and original late reverberation:

$$IR(t,\theta,\phi)_{dir} = IR(t,\theta,\phi)_{ERdir} + IR(t,\theta,\phi)_{LRorig} \tag{1}$$

where $IR(t,\theta,\phi)_{dir}$ is a pressure impulse response with associated directional information and directional weighting of early reflections. $IR(t,\theta,\phi)_{ERdir}$ refers to the early

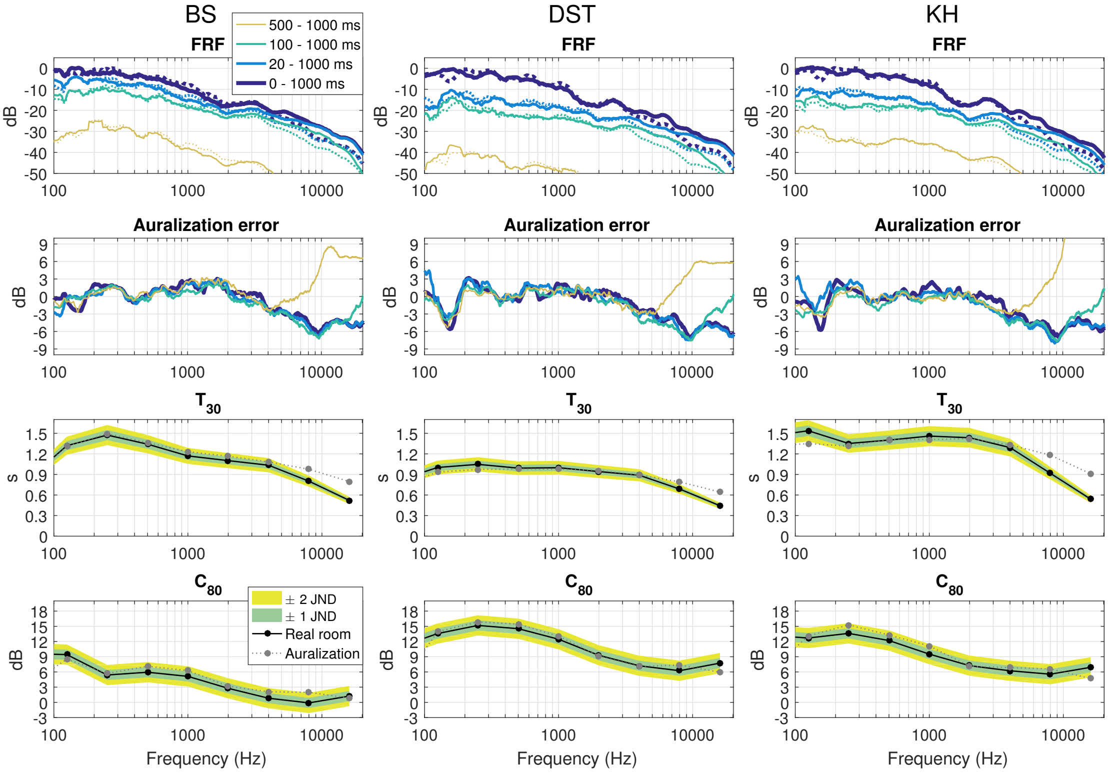


Figure 3: Comparison of frequency response (first row) and auralization error (second row) in different time intervals, and monaural room parameters (third and fourth row) of the measured rooms and their auralizations. Solid and dashed lines on the first row correspond to the real room and the auralization, respectively.

reflections after directional weighting and $IR(t,\theta,\phi)_{LRorig}$ is the late reverberation without modifications.

The two parts of the modified impulse response are:

$$IR(t,\theta,\phi)_{LRorig} = IR(t,\theta,\phi)_{orig} \cdot (1 - w(t)) \quad (2)$$

$$IR(t,\theta,\phi)_{ERdir} = IR(t,\theta,\phi)_{orig} \cdot g(\theta,\phi) \cdot w(t) \quad (3)$$

where $w(t)$ is a time window that defines the limit between early and late reverberation, including a mixing time between them. The term $g(\theta,\phi)$ refers to the spatial weighting function and is based on a cosine weighting in orthogonal directions. Five different cases are defined: all (no modification applied), front-back, sides, top-down, no-ER (early reflections removed completely):

$$g(\theta,\phi) = \begin{cases} 1 & \text{if all} \\ |\cos(\theta)\cdot\cos(\phi)| & \text{if front-back} \\ |\sin(\theta)\cdot\cos(\phi)| & \text{if sides} \\ |\sin(\phi)| & \text{if top-down} \\ 0 & \text{if no-ER} \end{cases} \quad (4)$$

The time window $w(t)$ is defined as

$$w(t) = \begin{cases} 1 & \text{if } t < t_{end} - t_{mix} \\ -1/t_{mix} & \text{if } t_{end} - t_{mix} \le t \le t_{end} \\ 0 & \text{if } t > t_{end} \end{cases} \quad (5)$$

where $w(t)$ is the time window, t_{end} is the start of late reverberation and t_{mix} is the mixing time.

3. EXPERIMENTS

The goal of the experiments is to explore the stage acoustic preferences of semi-professional trumpet players. To achieve this, two different experiments were carried out:

- General preference: This experiment tries to determine the room acoustical preferences of the players depending on the performance context.
- Directional Early Energy: This experiment examines the influence of the direction of the early reflections on the perceived stage support.

None of the experiments had a time limit, but the duration of every trial was measured for subsequent analysis. There were no requirements regarding the nature of the musical performance, instead, every musician was free to choose what to play in every experiment in order to explore the acoustic feedback of the rooms.

3.1 Procedure and aparatus

The experiments consisted on paired comparison tests in which all the stage acoustic conditions were compared against each other. The tests were conducted using a GUI presented on a screen close to the player. The GUI was developed in Max/MSP and was connected to the engine of the virtual environment [5], providing the information of the spatial impulse responses to be convolved in every trial. A trial consisted on the comparison of two rooms and the order of presentation of the trials was randomized. Musicians could switch in real time between the two acoustic

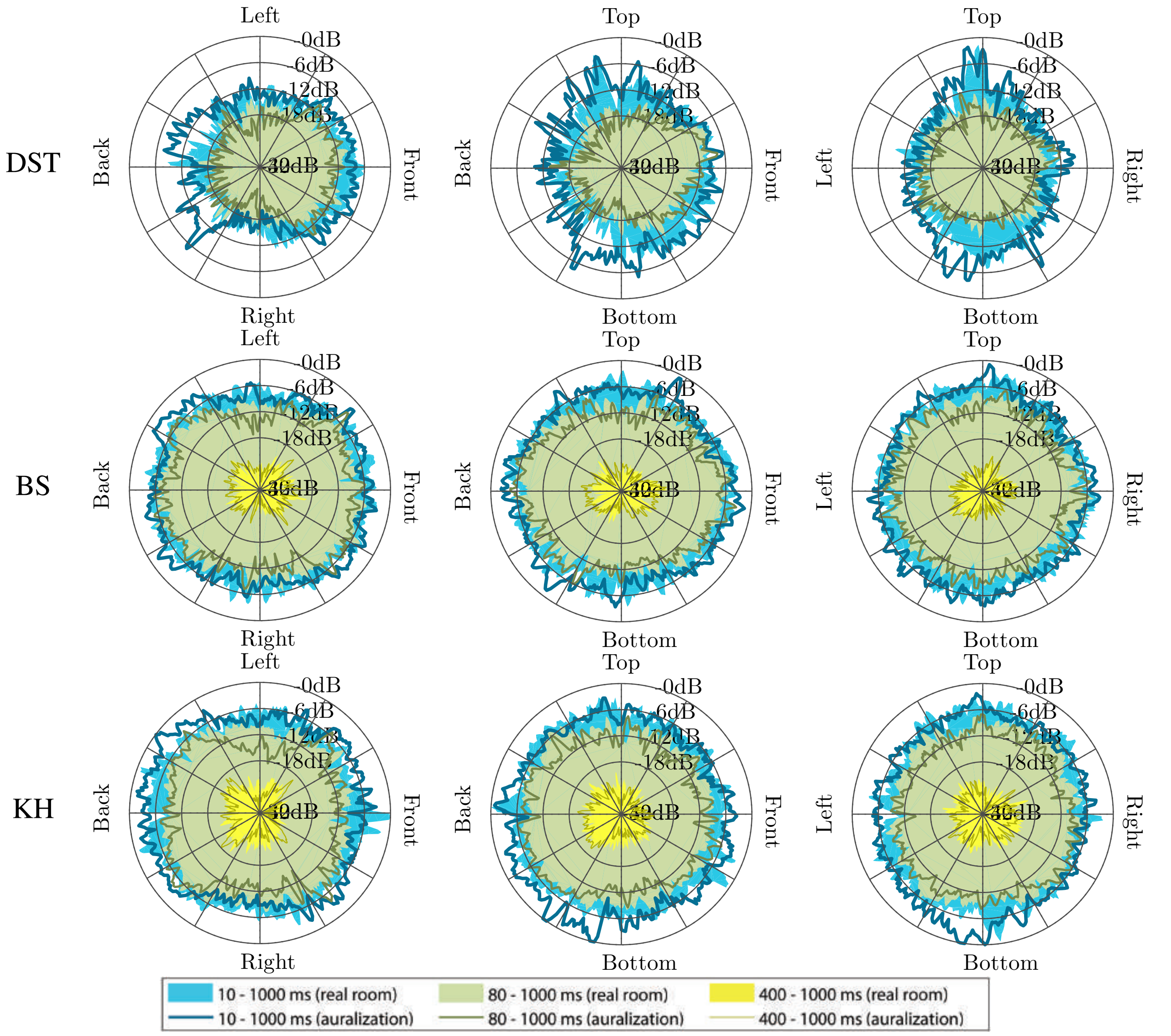


Figure 4: Broadband spatiotemporal representation of the auralized rooms (backward integrated) analyzed at the musician position.

situations and select the preferred choice in every trial by using a MIDI interface. The experimenter was present in the room during the test, but the interaction between the experimenter and the participants was minimized during the experiments to allow freedom in the judgments. An overview of the experimental set-up is presented in Fig. 5. All the participants were students of the Detmold University of Music at bachelor or master level, with an average age of 23 years at the time of the experiment. All of them had previously performed in concerts as soloists or with ensembles in the studied rooms.

3.2 General preference

Previous research has investigated musicians' stage acoustics preferences, usually focusing on overall impression or preferred acoustic conditions for concert performance. However, concert performance represents only a small fraction of time compared to the amount of time spent on practice and rehearsal activities. For this reason, five different

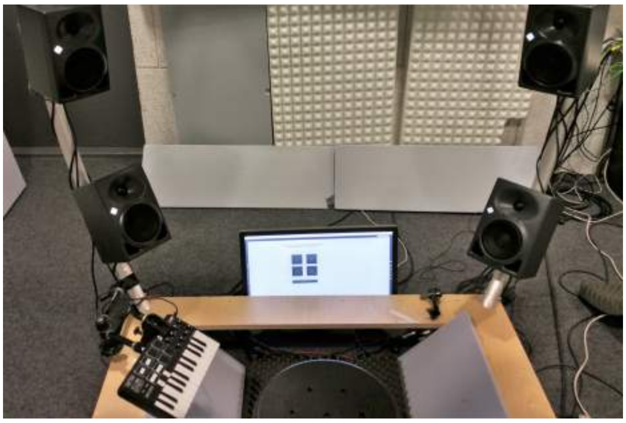


Figure 5: View of the set-up from the musician perspective.

tests are implemented, using the same methodology and acoustic conditions, but investigating five different case scenarios or performance contexts through a different question (in the following order):

- *Practice Technique*: In which room do you prefer to practice instrument technique?
- *Practice Concert*: Which room do you prefer to practice a concert piece?
- *Concert*: Which room do you prefer to perform in a concert?
- *Easiness*: In which room is it easier to perform?
- *Quality*: Which room provides the best overall acoustic (or sound) quality?

The experiment was completed by 7 different players and every section of the test lasted for approximately 10 minutes. The order of the trials was fully randomized, while the order of the test sections was always the same. Depending on the physical and psychological fatigue of the players the tests were completed in one or in two sessions. Five of the players completed a double pairwise comparison (5 tests x 6 pairs x 2 repetitions) with inverted orders to analyze the consistency of their answers. A preliminary analysis revealed a high consistency, thus to reduce the fatigue of the players, the last two players completed only a single pairwise comparison. The responses of these last two players were then duplicated to ensure an equal weight of all subjects in the analysis.

3.3 Directional Early Energy

The Early and Late Stage Support parameters provide an estimation for ensemble conditions and perceived reverberation on stage. They calculate the ratio between the direct sound (plus floor reflection), and a correspondent time interval (10 to 100 ms for early support, and 100 to 1000 ms for late support), and are formally defined as follows:

$$ST_{Early} = 10\log\left[\frac{\int_{0.02}^{0.1} p^2(t)dt}{\int_{0}^{0.01} p^2(t)dt}\right][dB] \qquad (6)$$

$$ST_{Late} = 10\log\left[\frac{\int_{0.1}^{1000} p^2(t)dt}{\int_{0}^{0.01} p^2(t)dt}\right][dB] \qquad (7)$$

where p is the instantaneous pressure of a measured impulse response on stage and t represents time in seconds. According to [2] and the correspondent ISO standard [12], stage parameters must be measured on stage, using an omnidirectional source and a distance of 1 meter between the source and the receiver. However, in the present study the measurement set-up consists of a directive source and a distance between source and receiver of approximately 62 cm. Although this set-up does not comply with the standard measurement requirements, it provides a more realistic approximation of the energy perceived by the musician, taking into consideration the real distance between instrument and performer and the radiation properties of the instrument.

The ability of hearing oneself and other musicians on stage is an important factor to facilitate ensemble performance and previous research suggests that the direction of arrival of early reflections on stage can influence the stage acoustics preferences of chamber orchestras [13], as well as solo musicians [4]. It is not clear, however, how this directionality should be quantified and how it is related to subjective perception of musicians on stage. The goal of the present study is to provide a methodology that allows a systematic study of those preferences in trumpet players using modified versions of the auralized rooms with different early energy levels and directions.

A pilot test was conducted with 5 participants, evaluating their preference in terms of stage support. The term was discussed with the participants to ensure a correct interpretation, and it was agreed that support is related to the ability of hearing their own sound in the room without difficulty or need to force their instrument (as described by Gade [2]).

The experiment followed the same procedure as in the general preference test. In this case there were 3 different rooms with 5 modifications per room (all, front-back, sides, top-down, no-ER). The order of the comparisons was randomized and all the participants rated each pair once, resulting in 30 comparisons (3 rooms x 10 pairs) per participant.

In all the studied halls the starting time of the late reverberation (t_{end}) was set to 100 ms, while the mixing time (t_{mix}) was 45, 45 and 65 ms for BS, DST and KH, respectively.

3.4 Interviews

After the experiments informal interviews were conducted with the musicians, asking them about their preferences, perceived differences between rooms and general opinion about the experiments. It is worth noting that the participants in the presented experiments already participated in previous research experiments [14], thus all of them were familiar with the rooms and their auralized versions. The interviews were based on a cooperative conversation, in an attempt to focus on the aspects that were more important for musicians. A written interview form was provided as well, to provide the opportunity to comment on aspects not present in the conversation.

4. RESULTS

The results of the paired comparisons were processed with a Bradley-Terry-Luce (BTL) model, which estimates the probability of preference of every room in each scenario by comparing the result of all the direct comparisons [15]. Tables 1 and 2 show the average and median times per comparison in the tests.

4.1 General preference

The estimated preference in every studied case is presented in Fig 6. The results show that the performance context is determinant when choosing a certain kind of stage acoustics over another. Drier rooms are preferred to practice in-

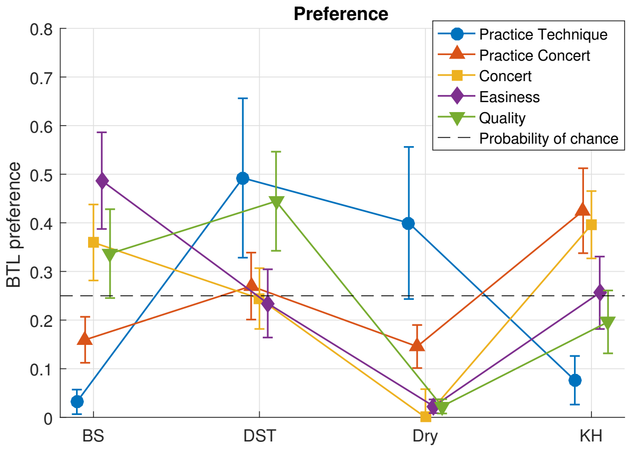


Figure 6: Estimated preference of different rooms depending on the playing purpose.

strument technique, being DST and anechoic conditions the most preferred. The most preferred room to practice a concert (KH) is as well the most preferred room to perform in a concert, and while BS is the second most preferred room to perform in a concert, there is no statistical difference between their preference. The ease of performance is dominated by the loudest room (BS), which together with DST are the best rated rooms in terms of overall sound quality.

In other words, every room stands out in a certain kind of performance context:

- BS: *Concert, Easiness, Quality*.
- DST: *Practice Technique, Quality*.
- Dry: *Practice Technique*.
- KH: *Practice Concert, Concert*.

4.1.1 Preference and acoustic parameters

Diverse room acoustic parameters have been extracted from the measured auralizations: reverberation time (RT$_{20}$) and room gain parameters (G$_{all}$, G$_{early}$ and G$_{late}$). The reason for using room gain parameters is that the values of ST$_{early}$ and ST$_{late}$ in anechoic conditions are -∞, thus not allowing curve fitting operations. The parameters are defined as follows:

$$G_{all} = 10 \log \left[\frac{\int_0^{\infty} p^2(t)dt}{\int_0^{0.01} p^2(t)dt} \right] [dB] \quad (8)$$

$$G_{early} = 10 \log \left[\frac{\int_0^{0.1} p^2(t)dt}{\int_0^{0.01} p^2(t)dt} \right] [dB] \quad (9)$$

$$G_{late} = 10 \log \left[\frac{\int_0^{0.01} p^2(t)dt + \int_{0.1}^{\infty} p^2(t)dt}{\int_0^{0.01} p^2(t)dt} \right] [dB] \quad (10)$$

where p is the instantaneous pressure of the impulse response and t is time in seconds. The gain parameters have been computed in frequency bands and averaged over octave bands from 125 Hz to 2000 Hz. The reverberation time is an average of the bands 500 Hz and 1000 Hz.

	Avg. dur. (s)	Median dur. (s)
Pract. Technique	79	42
Pract. Concert	45	35
Concert	33	28
Easiness	41	31
Quality	40	31

Table 1: Duration of trials for the general preference experiment

	Avg. dur. (s)	Median dur. (s)
DST	38	28
BS	45	39
KH	40	27

Table 2: Duration of trials for the directional early energy experiment

Considering a previous study that suggested a possible quadratic relationship between early energy and subjective overall impression [16], a quadratic curve fitting has been applied between the room parameters and the average preferences of every case. The fittings are depicted in Fig 7, and only curves with an adjusted R^2 > 0.6 are presented. Although the context *Practice Technique* is better suited for drier and quieter rooms, there is no clear fitting between the parameters and the different performance contexts. *Practice concert* presents a quadratic relationship with the late gain of the room (G$_{late}$), while *Concert* presents a quasi linear relationship with RT$_{20}$ and clear quadratic trends with the rest of the parameters. *Easiness* presents a quadratic relationship with G$_{all}$ and seems to benefit from a strong early and overall energy (quasi linear relationship with G$_{early}$ and G$_{all}$ for the studied cases). Finally, *Quality* presents a quadratic relationship with RT$_{20}$.

4.2 Directional Early Energy

The BTL estimated preferences regarding the direction of early energy are depicted in Fig. 8. The room DST presents a statistically significant higher preference when all the early energy is present, when compared to any other case. For the rooms BS and KH, although *Front-back* energy is slightly more preferred and *no-ER* is the least preferred case, the differences are not statistically significant.

Comparing the estimated preferences against the ST$_{early}$ values of the modified early energy auralizations it appears that when a certain value of support (ST_{early}) is met (from approximately -12 dB and upwards) the amount of early energy or the direction of it are irrelevant (see Fig. 9). This would suggest that the level of the early energy is more important for solo musicians than the direction of arrival of this energy. However, the trends are fairly weak and seem to be room dependent, thus more experiments are needed to draw clear conclusions.

5. DISCUSSION

5.1 Comments on the results

During interviews, musicians often rated anechoic characteristics (Dry room) to be very difficult to play in, un-

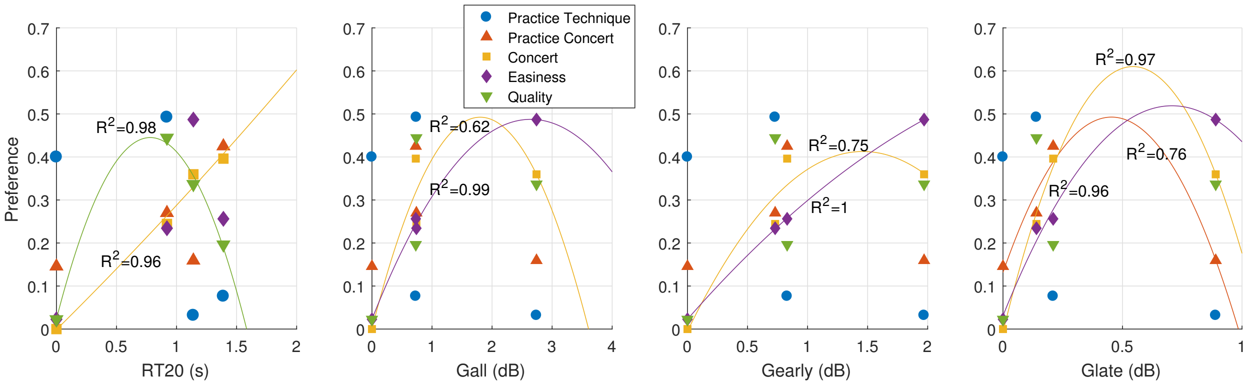


Figure 7: Quadratic fits of the room preferences and room acoustic parameters. The R^2 value included in the graph corresponds to the adjusted R^2.

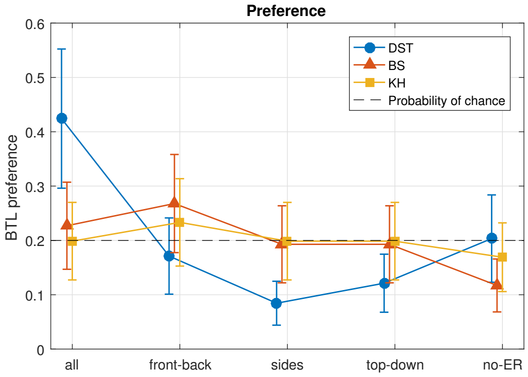


Figure 8: Estimated preference of early energy directions.

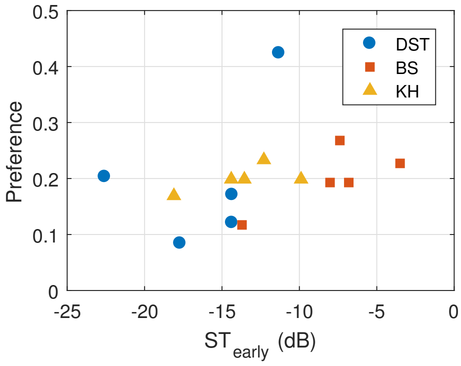


Figure 9: Preference values of directional energy in relation to ST_{early} values.

comfortable and more exhausting. However, in such cases where the playing requires a high level of concentration and ability to hear own mistakes, a very dry room is often preferred. Indeed, a small amount of reverberation is beneficial in keeping a high clarity and reduce the fatigue of the players. When practicing for a concert, the preference shifts towards slightly longer reverberation times and louder rooms, which could be closer to concert conditions. However, it is necessary to retain a certain degree of clarity or early energy, in order allow the detection and correction of musical performance errors or articulation imperfections. When playing in a concert, musicians often prefer longer reverberation times, which could be related with a bigger projection of the instrument and the feeling of playing in front of a bigger audience. The easiness of playing, or musician comfort is directly related with the amount of early energy, as musicians feel more supported by the stage conditions and less effort is needed to develop the notes and feel their presence in the room. During the interviews multiple participants stated that when they feel supported by the room it is easier to relax, reduce fatigue and ultimately perform better. Finally, although in the studied rooms *Quality* seems to be quadratically related with RT_{20}, it has to be considered that more aspects, such as tone balance, are involved and purely energetic terms are insufficient to rate the sound quality of a room.

In addition, it is interesting that during the different parts of the first test (general preference), the played excerpts varied substantially depending on the test question. For *Practice Technique* musicians often tended to play fast passages with complex note articulations, meaning that they tended to explore the early response of the room and its clarity. *Practice concert* and *Concert* were usually related with more lyrical passages. However, those technical and lyrical passages could be in many cases parts of the same piece. Additionally, when testing *Easiness* and *Sound Quality* they often played a broad type of pieces with different characters, testing the articulation and judging the tone color and the decay of the rooms. Finally, the trial duration (see Tab. 1) of the *Practice Technique* case is significantly higher than in the other cases ($p < 0.01$), which could indicate that there is a learning effect in judging the different rooms and the task becomes easier when musicians are more familiar with the different conditions.

Regarding the test of directional early energy, most of the musicians commented that in many cases it is very difficult to distinguish which are the differences of the rooms. In fact, most of the players tended to play notes with clear articulation to explore the early part of the sound. As seen in the results, the direction of arrival of the early energy seems to not be relevant to judge their preference, instead the level of the early energy would be more important. It is thus not clear, what would be the directional preference in case of having equal levels in the different cases. In future experiments the effect of the level should be considered in order to include a proper equalization of early energy. In addition, the length of the window and mixing time should be carefully decided to provide a wide and realistic variety of acoustic scenarios. This pilot study demonstrated the potential of manipulating directional energy in virtual environments, allowing a detailed control of the acoustic scene and enabling a systematic study of different stage acoustic phenomena.

5.2 Comparability with previous studies

Previous research on stage acoustics for solo musicians usually studied musicians' *preference* of stage acoustics, which intuitively translates to preference to perform in a concert. However, *preference* could be understood as well as sound quality or degree of comfort provided by the room. The present study demonstrates that the same musicians judging the same rooms lead to very different results depending on the formulation of the question and judging criteria or performance context. This makes it very difficult to compare the results of the present study with previous research. However, if we compare the concert conditions with previous results of *preference* the results support former findings. Here, a quasi linear relationship was found between RT_{20} and the preference of a room to perform in a concert and longer reverberation times were also preferred in previous studies (up to 2.3 s [4] and around 1.9 s in [3]). However, the longest (and most preferred) reverberation time in the present test is approximately 1.4 s.

The comparison of stage energy parameters, such as support (ST) or strength (G) depends indeed on the distance between the source and the receiver and the directivity of the source. In addition, while they have been standardly measured in previous research [2, 3], the calibration procedures, instrument directivity and micking techniques applied in the virtual environments leave some room for uncertainties on the absolute values and it is not straightforward to ensure that the same values measured in different virtual environments are equivalent.

6. CONCLUSION

Two experiments on musicians' stage preferences are presented. The results demonstrate that the performance context is a key factor for musicians to prefer a particular stage acoustics. Moreover, the direction of the early energy does not seem to play an important role on determining the preference of stage support, while the level of the early energy is more important. The results seem to support partially previous findings, although the comparability of the results depends greatly on the measurement process and calibration process of the virtual environments.

Directions of future work should increase the range of tested rooms and instruments, as well as a careful design of the directional properties of the auralized rooms.

Acknowledgments

The authors want to acknowledge the musicians participating in the experiments for their time and effort. The research work presented in this article has been funded by the European Commission within the ITN Marie Curie Action project BATWOMAN under the 7th Framework Programme (EC grant agreement no. 605867).

7. REFERENCES

[1] S. V. Amengual Garí, W. Lachenmayr, and M. Kob, "Study on the influence of acoustics on organ playing using room enhancement," in *Proceedings of the Third Vienna Talk on Music Acoustics*, Vienna, Austria, 2015, pp. 252–258.

[2] A. C. Gade, "Investigations of musicians' room acoustic conditions in concert halls. part I: Methods and laboratory conditions," *Acustica*, vol. 69, pp. 193–203, 1989.

[3] K. Ueno and H. Tachibana, "Experimental study on the evaluation of stage acoustics by musicians using a 6-channel sound simulation system," *Acoust. Sci. & Tech*, vol. 24, no. 3, pp. 130–138, 2003.

[4] A. Guthrie, S. Clapp, J. Braasch, and N. Xiang, "Using ambisonics for stage acoustics research," in *International Symposium on Room Acoustics*, Toronto, Canada, 2013.

[5] S. V. Amengual Garí, D. Eddy, M. Kob, and T. Lokki, "Real-time auralization of room acoustics for the study of live music performance," in *Fortschritte der Akustik - DAGA 2016*, Aachen, Germany, 2016, pp. 1474–1777.

[6] S. Tervo, J. Pätynen, A. Kuusinen, and T. Lokki, "Spatial decomposition method for room impulse responses," *J. Audio Eng. Soc.*, vol. 61, no. 1/2, pp. 16–27, January/February 2013.

[7] V. Pulkki, "Virtual sound source positioning using vector base amplitude panning," *J. Audio Eng. Soc.*, vol. 45, pp. 456–466, 1997.

[8] J. Pätynen, S. Tervo, and T. Lokki, "Analysis of concert hall acoustics via visualizations of time-frequency and spatiotemporal responses," *J. Acoust. Soc Am.*, vol. 133, no. 2, pp. 842–857, Feb. 2013.

[9] S. Tervo, J. Pätynen, N. Kaplanis, M. Lydolf, S. Bech, and T. Lokki, "Spatial analysis and synthesis of car audio system and car cabin acoustics with a compact microphone array," *J. Audio Eng. Soc.*, vol. 63, no. 11, pp. 914–925, 2015.

[10] S. Tervo and J. Pätynen. SDM toolbox. [Online]. Available: http://de.mathworks.com/matlabcentral/fileexchange/56663-sdmtoolbox

[11] J. Pätynen, S. Tervo, and T. Lokki, "Amplitude panning decreases spectral brightness with concert hall auralizations," in *Audio Engineering Society Conference: 55th International Conference: Spatial Audio*, Aug 2014.

[12] *Acoustics – Measurement of room acoustic parameters – Part 1: Performance spaces*, Std. ISO 3382-1:2009, 2009.

[13] L. Panton, D. Holloway, D. Cabrera, and L. Miranda, "Stage acoustics in eight australian concert halls: Acoustic conditions in relation to subjective assessments by a touring chamber orchestra," *Acoustics Australia*, pp. 1–15.

[14] S. V. Amengual Garí, T. Lokki, and M. Kob, "Live performance adjustments of solo trumpet players due to acoustics," in *International Symposium on Musical and Room Acoustics (ISMRA)*, La Plata, Argentina, 2016.

[15] F. Wickelmaier and C. Schmid, "A matlab function to estimate choice model parameters from paired-comparison data," *Behavior Research Methods, Instruments, and Computers*, vol. 1, no. 36, pp. 29–40, 2004.

[16] W. Chiang, C. Shih-tang, and H. Ching-tsung, "Subjective assessment of stage acoustics for solo and chamber music performances," *Acta Acustica united with Acustica*, vol. 89, no. 5, pp. 848–856, 2003.

Exploring Social Mobile Music with Tiny Touch-Screen Performances

Charles P. Martin
Department of Informatics,
University of Oslo, Norway
charlepm@ifi.uio.no

Jim Torresen
Department of Informatics,
University of Oslo, Norway
jimtoer@ifi.uio.no

ABSTRACT

Touch-screen musical performance has become commonplace since the widespread adoption of mobile devices such as smartphones and tablets. However, mobile digital musical instruments are rarely designed to emphasise collaborative musical creation, particularly when it occurs between performers who are separated in space and time. In this article, we introduce an app that enables users to perform together asynchronously. The app takes inspiration from popular social media applications, such as a timeline of contributions from other users, deliberately constrained creative contributions, and the concept of a reply, to emphasise frequent and casual musical performance. Users' touch-screen performances are automatically uploaded for others to play back and add reply performances which are layered as musical parts. We describe the motivations, design, and early experiences with this app and discuss how musical performance and collaboration could form a part of social media interactions.

1. INTRODUCTION

Popular social media apps for mobile devices have allowed millions of users to engage with creative production of images and text. These devices' cameras, touch-screens, powerful processors, and portability suggest on-the-go creativity, and it would appear that straightforward sharing with friends, or a wider network of followers, is a key factor in encouraging users to create content of all forms. Given the many affordances of mobile devices, it has been well noted that they are suitable platforms for mobile music making [1]. Despite many creative mobile digital musical instruments (DMIs) appearing in recent years, we have yet to see the widespread adoption of musical creation as an integrated element of social media. Furthermore, few musical apps have attempted to emphasise ensemble, rather than individual, performance, even though group music-making is often seen as a valuable social activity.

In this article, we present the design for *MicroJam*[1] [2], a collaborative and social mobile music-making app. This design emphasises casual, frequent, and social performance. As shown in Figure 1, the app features a very simple touch-screen interface for making electronic music where skill is not a necessary prerequisite for interaction. Performances are limited to five seconds and uploaded automatically to encourage improvisation and creation rather than editing. Users can reply to others' performances by recording a new layer, this combines social interaction with musical ensemble interaction. In sections 1.1 and 1.2, we will motivate MicroJam's design with a discussion of music-making in social media, and the possibilities for asynchronous and distributed collaborations with mobile musical interfaces. In Section 2 we will describe the app in detail and, by way of a preliminary evaluation, we will explore the performance contributions of early users and testers of the app.

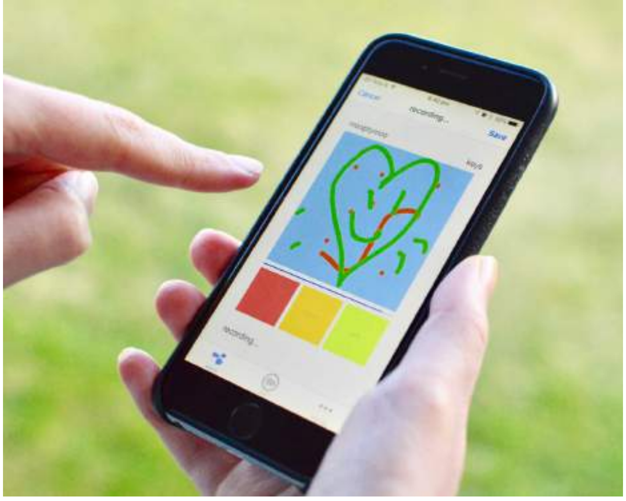


Figure 1. MicroJam allows users to create tiny touch-screen performances that are shared to other users. Replies to performances form distributed and asynchronous duets. A video demo of MicroJam is available online: https://youtu.be/SkUjjQd13KU

1.1 Social Media and Music-Making

Many social media platforms emphasise the value of concise and frequent contributions by users. Twitter famously limits written notes to 140 characters. Instagram used a square image format and film-like processing to emphasise the creative possibilities of mobile phone images. Both of these services show users a timeline of contributions from others that they follow, who might be friends, celebrities, or just interesting strangers. Posts in these services are in-

[1] Source code and further information about MicroJam is available online at DOI:10.5281/zenodo.322364

Same Location	Mobile Phone Orchestra [10]; Ensemble Metatone [11] Viscotheque Jams [12].	Locative Performance
Different Location	Networked Musical Performance Magic Piano [14]	Glee Karaoke [13]; MicroJam Performances.
	Same Time	**Different Time**

Table 1. Ensemble performances typically occur with all participants in the same time and place; however, other situations are possible. MicroJam focuses on performances that are distributed (different place) and asynchronous (different time).

tended to be frequent, casual, and ephemeral. Snapchat introduced an interface that optimised replying to many image messages in one session. This app further emphasised ephemerality by introducing images that expire and can only be viewed for a few seconds after being opened. These and other social applications have attracted many millions of users and encouraged their creativity in the written word and photography. While social media is often used to promote music [3], music *making* has yet to become an important creative part of the social media landscape.

While music is often seen as an activity where accomplishment takes practice and concerted effort, casual musical experiences are well-known to be valuable and rewarding creative activities. Accessible music making, such as percussion drum circles, can be used for music therapy [4]. Digital Musical Instruments (DMIs) such as augmented reality instruments [5] and touch-screen instruments [6] have also been used for therapeutic and casual music-making. In the case of DMIs, the interface can be designed to support creativity of those without musical experience or with limitations on motor control. Apps such as *Ocarina* [7] and *Pyxis Minor* [8] have shown that simple touch-screen interfaces can be successful for exploration by novice users as well as supporting sophisticated expressions by practised performers.

Some mobile music apps have included aspects of social music-making. Smule's *Leaf Trombone* app introduced the idea of a "world stage" [9]. In this app, users would perform renditions of well-known tunes on a unique trombone-like DMI. Users from around the world were then invited to critique renditions with emoticons and short text comments. World stage emphasised the idea that while the accuracy of a rendition could be rated by the computer, only a human critic could tell if it was ironic or funny. Indeed, *Leaf Trombone*, and other Smule apps have made much progress in integrating musical creation with social media.

1.2 Jamming through Space and Time

While performance and criticism is an important social part of music-making, true musical collaboration involves performing music together. These experiences of group creativity can lead to the emergence of qualities, ideas, and experiences that cannot be easily explained by the actions of the individual participants [15]. Mobile devices have often been used in ensemble situations such as *MoPho* (Stanford Mobile Phone Orchestra) [10], *Pocket Gamelan* [16], and *Ensemble Metatone* [11]; however, in these examples, the musicians played together in a standard concert situation.

Given that mobile devices are often carried by users at all times, it would be natural to ask whether mobile device ensemble experiences can be achieved even when performers are not in a rehearsal space or concert venue. Could users contribute to ensemble experiences at a time and place that is convenient to them? The use of computer interfaces to work collaboratively even when not in the same space and time has been extensively discussed. In HCI, groupware systems have been framed using a time-space matrix to address how they allow users to collaborate in the same and different times and places [17]. For many work tasks, it is now common to collaborate remotely and at different times using tools such as Google Docs or Git; however, distributed and asynchronous musical collaboration is not as widely accepted.

In Table 1, we have applied the time-space matrix to mobile musical performance. Conventional collaborative performances happen at the same time and location. Even with mobile devices, most collaboration has occurred in this configuration. Collaborations with performers distributed in different locations but performing at the same time are often called networked musical performances [18]. Early versions of Smule's *Magic Piano* [14] iPad app included the possibility of randomly assigned, real-time duets with other users. Networked performances are also possible with conventional mobile DMIs and systems for real-time audio and video streaming.

Performance with participants in different times, the right side of Table 1, are less well-explored than those on the left. Performances where participants are in the same place, but at different times, could come under the banner of locative performances, such as *Net_dérive* [19] or *Sonic City* [20], where geographical location is an important input to a musical process.

The final area of the matrix involves music-making with performers that are in different places and different times. *Glee Karaoke* [13] allows users to upload their sung renditions of popular songs, and add layers to other performers' contributions. The focus in this app, however, is on singing along with a backing track, and the mobile device does not really function as a DMI but an audio recorder. These limitations rule out many musical possibilities; for instance, touch-screen DMIs can create a variety of sounds from similar interfaces, so orchestras of varying instruments could be assembled from many remote participants to improvise original music. It remains to be seen whether large scale musical collaboration between distributed users is possible. It seems likely, however, that such collaborations would uncover hidden affordances of the medium as has been seen in other distributed online media such as "Twitch Plays Pokémon" [21]. Our app, MicroJam, also fits into this lower-right quadrant. In the next section, we will describe how this new app enables distributed and asynchronous collaboration on original musical material.

2. MICROJAM

MicroJam is an app for creating, sharing, and collaborating with tiny touch-screen musical performances. This app has been specifically created to interrogate the possibilities for collaborative mobile musical performances that span space and time. While these goals are lofty, the design of MicroJam has been kept deliberately simple. The main screen recalls social-media apps for sharing images. Musical performances in MicroJam are limited to very short interactions, encouraging frequent and ephemeral creative contribution. MicroJam is an iOS app written in Swift and uses Apple's CloudKit service for backend cloud storage. The source code is freely available for use and modification by other researchers and performers [2]. In the following sections we will discuss the design of the app, the format of the tiny musical performances that can be created, and some early experiences and musical performances recorded with users.

2.1 Design

The primary screen for MicroJam is the performance interface, shown in the centre of Figure 2 and called *jam!*. This screen features a square touch-area which is initially blank. Tapping, swirling, or swiping anywhere in this area will create sounds and also start recording touch activity. All touch interaction in this area is visualised with a simple paint-style drawing that follows the user's touches, and is simultaneously sent to a Pure Data patch running under `libpd` [22] to be sonified. After five seconds of touch interaction, the recording is automatically stopped (although the performer can continue to interact with the touch area). The recording can be subsequently replayed with the *play* button, or looped with the *jam* button.

Users of MicroJam can choose the sound-scheme used to sonify their interactions in the jam interface. At present, four options are available through the settings screen: *chirp*, a simple theremin-like sound with pitch mapped to the x-dimension of interactions; *keys*, a Rhodes-like keyboard sound with pitch and timbre mapped to the x- and y-dimensions; *strings*, a simple modelled string sound that performs mandolin rolls and changes timbre as the performer moves around the screen; and *drums*, a simple drum machine with different sounds mapped to quadrants of the screen. As each of these sound-schemes is implemented as a Pure Data patch, it is straightforward to add more to MicroJam in future.

Previously recorded performances, and those recorded by other users and downloaded from the server, are listed in the *world* screen as shown on the left side of Figure 2. Each performance is represented by a visual trace of the touch-drawing captured during recording as well as the name of the contributor and date of the performance. Any one of these performances can tapped which opens the recording back up in the jam screen for playback. When playing back, both the sound and visualised touch-interactions are replayed in the touch-area.

When viewing a previously saved performance, the user can tap *reply*, to open a new layer on top of the recording. As shown in the right side of Figure 2, the previous as well as current touch-visualisations are shown as well as separate sonifications for each layer. At present, only one reply is possible in MicroJam; however, the ability to reply to replies, creating multi-layered performances is planned for future versions.

2.2 Tiny touch-screen performances

MicroJam is intended to provide a musical experience that is similar to other mobile social creativity applications. One innovation in this field has been to constrain contributions, leading to more frequent interactions and possibly higher creativity due to the lower stakes and effort. Musical interactions in MicroJam are similarly constrained to be tiny touch-screen performances: those that are limited in the area and duration of interaction. We define a *tiny* performance as follows:

1. All touches take place in a square subset of the touch-screen.
2. Duration of the performance is five seconds.
3. Only one simultaneous touch is recorded at a time.

Such performances require very little effort on the part of users. While some users may find it difficult to conceive and perform several minutes of music on a touch-screen device, five seconds is long enough to express a short idea, but short enough to leave users wanting to create another recording. It has been argued that five seconds is enough time to express a sonic object and other salient musical phenomena. [23]. While the limitation to a single touch may seem unnecessary on today's multi-touch devices, this stipulation limits tiny performances to monophony. In order to create more complex texture or harmony, performers must collaborate, or record multiple layers themselves.

For transmission and long-term storage, tiny touch-screen performances are stored as simple comma-separated values files. The data format records each touch interactions time (as an offset in seconds from the start of the performance), whether the touch was moving or not, x and y locations, as well as touch pressure. In MicroJam, the visual trace of performances is also stored for later use in the app, although this could be reconstructed from the touch data.

2.3 User Data and Early Experiences

The prototype version of MicroJam has been distributed and demonstrated to small numbers of researchers and students. These early experiences have allowed us to streamline the creative workflow of the app and to observe the kind of tiny performances that can be created in the jam interface. Since the release of the first prototype, around 200 tiny performances have been collected. Most of the

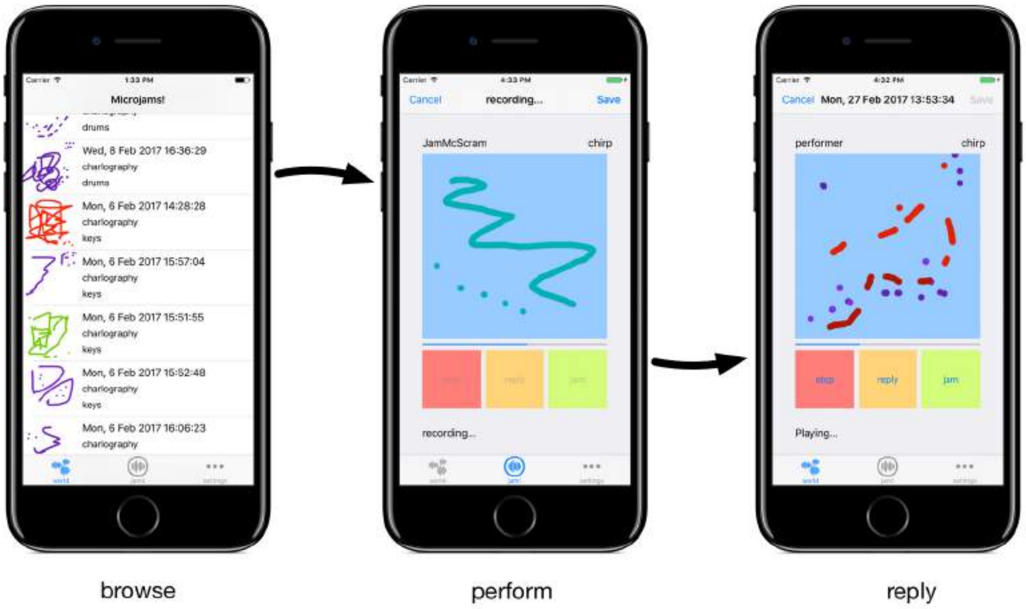


Figure 2. The MicroJam interface allows users to browse a timeline of performances (left), create new performances (centre), and reply, or play in duet, with previously recorded performances (right).

participants were computer science and engineering students with little computer music experience, a few music technology students were also included.

The visualisations of a subset of these performances are reproduced in Figure 3. This figure shows the variety of touch-interaction styles that have already been observed in performers. Many of the interactions are abstract, resembling scribbles that show the user experimenting with the synthesis mapping of the jamming interface. In some performances, repeated patterns can be seen, where performers have repeated rhythmic motions in different parts of the touch-area. A number of the performances are recognisable images: figures, faces, and words. These users were particularly interested in how the interface maps between drawing, a visual and temporal activity, and sound. Observing the users, it seems that some focused on what a particular drawing might *sound* like, while others were interested in what particular sounds *look* like. At present, these performance recordings have not been analysed with respect to which sound-scheme was in use, and how reply layers fit together. Further experiments could seek to understand these relationships.

It has been gratifying to hear that several early users of the app greatly enjoyed the experience of creating tiny performances, and wished that similar interactions could be integrated into existing social apps. These users immediately set about recording multiple performances, exploring the sound-schemes and the creative possibilities of the touch-screen interface. Other users, however, had a lukewarm reaction to the concept of free-form touch-screen musical performance. It could be that casual users do not expect to be able to create original music. After all, musical performance is often (erroneously, we feel) seen as a task only for specialists with a high level of training or talent. It may be a hard sell to ask users to create their own music, and to collaborate with others. Rather than a discouragement, we see this as an opportunity to continue developing mobile music experiences that push the boundaries of everyday music making. Understanding how comfortable users would be composing original tiny performances could be addressed in future studies. The app design could also include more guidance, such as a training system, or more extensive visual feedback, to help users who are unsure about making touch-screen music.

3. CONCLUSIONS AND FUTURE WORK

In this paper we have advocated for social apps for creating music, as opposed to more popular written and visual media. We have argued that such apps could take advantage of the ubiquity of mobile devices by allowing users to collaborate asynchronously and in different locations, and shown that such modes of interaction are relatively unexplored compared to more conventional ensemble performances. Our app, MicroJam, represents a new approach to asynchronous musical collaboration. Taking inspiration from the ephemeral contributions that typify social media apps, MicroJam limits performers to tiny five-second touch-screen performances, but affords them extensive opportunities to browse, playback, and collaborate through responses. MicroJam's tiny performance format includes a complete listing of the touch interactions as well as a simple visualisation of touch interactions. This format allows performances to be easily distributed, viewed, and studied. Early experiences with MicroJam have shown that users can engage with the interface to create a range of interesting performances. While some see potential to include music-making in their social media activities, others may lack confidence about producing music, even in the tiny performance format.

There is much scope for refinement and development of

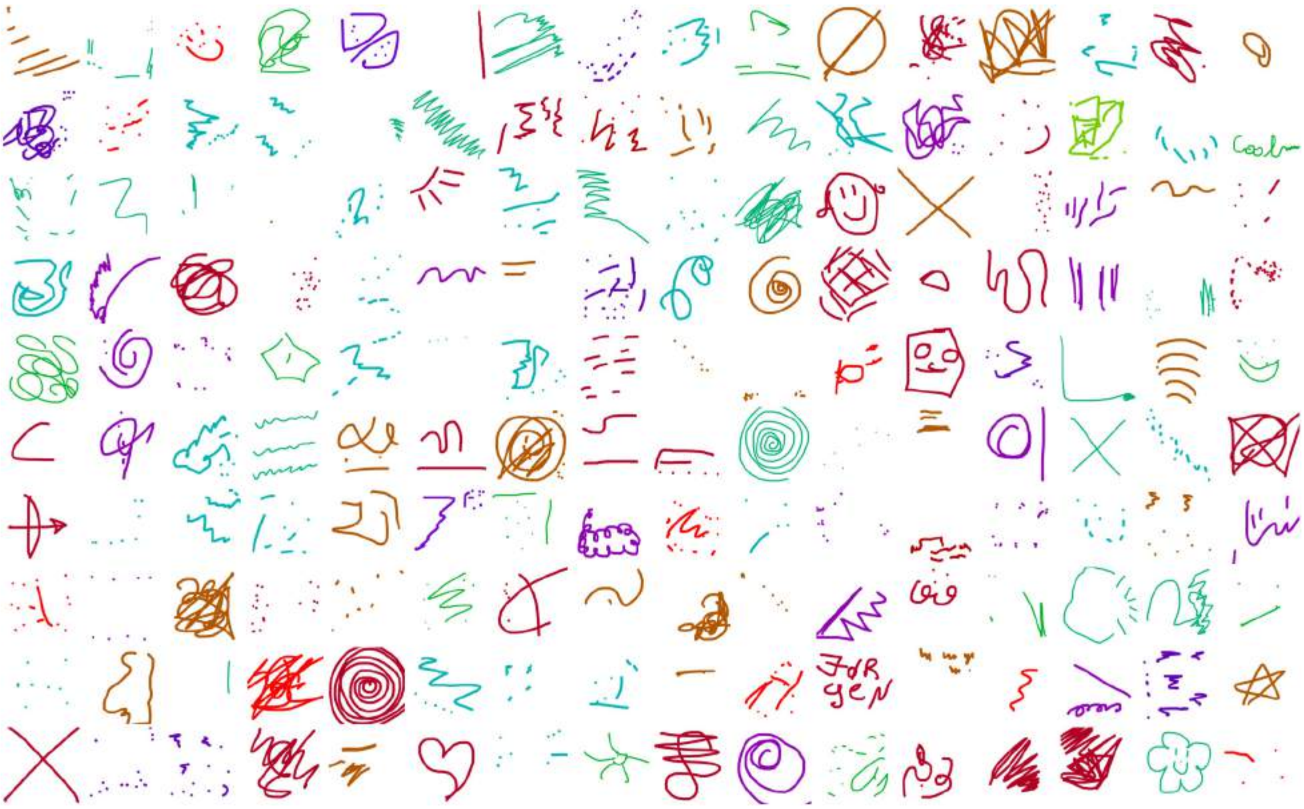


Figure 3. The visual trace of performances in the prototype for MicroJam. Some performances focus on similar touch gestures in different locations of the jamming area, others appear to be exploratory scribbles, and several focus on visual interaction.

MicroJam in future work. Enhancements such as more varied and refined sound-schemes and visualisations could be appealing to users. Future efforts could also focus on enhancing distributed collaborations. Allowing more than one reply to MicroJam performances could generate very large collaborations between users. If multiple reply threads were available, users might be able to generate complex performance structures. Automatic traversal of such structures could constitute a kind of generative composition with users' original musical material. As MicroJam affords a high quantity of short interactions, data collected from the app could be used to train generative models for tiny performances. It may be possible to predict potential replies to a given performance or to generate performances that extend beyond five seconds while keeping within a user's style. Exploring these interactions with experienced, as well as novice, musicians could point the way to more expressive and musically powerful interactions in MicroJam.

Integrating music-making, as opposed to the more conventional music appreciation, or music promotion, into social media calls into question the musical confidence and creative aspirations of users. Future studies could examine how users could potentially include mobile music-making in everyday social media interactions. The precedent set by other successful mobile music apps suggest that users do seek out musical outlets for their creativity. Future work with MicroJam may focus on guiding beginner users towards more musical confidence and rewarding their exploratory improvisations.

Acknowledgments

This work is supported by The Research Council of Norway as a part of the Engineering Predictability with Embodied Cognition (EPEC) project, under grant agreement 240862.

4. REFERENCES

[1] A. Tanaka, "Mapping out instruments, affordances, and mobiles," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, ser. NIME '10, K. Beilharz, A. Johnston, S. Ferguson, and A. Y.-C. Chen, Eds. Sydney, Australia: University of Technology Sydney, 2010, pp. 88–93. [Online]. Available: http://www.nime.org/proceedings/2010/nime2010_088.pdf

[2] C. P. Martin, "Microjam v0.1-alpha," Git Repository, Feb. 2017. [Online]. Available: https://doi.org/10.5281/zenodo.322364

[3] S. Dewan and J. Ramaprasad, "Social media, traditional media, and music sales." *MIS Quarterly*, vol. 38, no. 1, pp. 101–121, 2014.

[4] S. Scheffel and B. Matney, "Percussion use and training: A survey of music therapy clinicians," *Journal of Music Therapy*, vol. 51, no. 1, p. 39, 2014. DOI: 10.1093/jmt/thu006

[5] A. G. D. Correa, I. K. Ficheman, M. d. Nascimento, and R. d. D. Lopes, "Computer assisted music therapy: A case study of an augmented reality musical system for children with cerebral palsy rehabilitation," in *International Conference on Advanced Learning Technologies*. IEEE, 2009. DOI:10.1109/ICALT.2009.111 pp. 218–220.

[6] S. Favilla and S. Pedell, "Touch screen ensemble music: Collaborative interaction for older people with dementia," in *Proceedings of the 25th Australian Computer-Human Interaction Conference*, ser. OzCHI '13. New York, NY, USA: ACM, 2013. DOI:10.1145/2541016.2541088 pp. 481–484.

[7] G. Wang, "Ocarina: Designing the iPhone's magic flute," *Computer Music Journal*, vol. 38, no. 2, pp. 8–21, 2014. DOI:10.1162/COMJ_a_00236

[8] T. Barraclough, D. Carnegie, and A. Kapur, "Musical instrument design process for mobile technology," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, E. Berdahl and J. Allison, Eds. Baton Rouge, Louisiana, USA: Louisiana State University, May 2015, pp. 289–292. [Online]. Available: http://www.nime.org/proceedings/2015/nime2015_313.pdf

[9] G. Wang, S. Salazar, J. Oh, and R. Hamilton, "World stage: Crowdsourcing paradigm for expressive social mobile music," *Journal of New Music Research*, vol. 44, no. 2, pp. 112–128, 2015. DOI:10.1080/09298215.2014.991739

[10] J. Oh, J. Herrera, N. J. Bryan, L. Dahl, and G. Wang, "Evolving the mobile phone orchestra," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, ser. NIME '10, K. Beilharz, A. Johnston, S. Ferguson, and A. Y.-C. Chen, Eds. Sydney, Australia: University of Technology Sydney, 2010, pp. 82–87. [Online]. Available: http://www.nime.org/proceedings/2010/nime2010_082.pdf

[11] C. Martin, H. Gardner, and B. Swift, "Tracking ensemble performance on touch-screens with gesture classification and transition matrices," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, ser. NIME '15, E. Berdahl and J. Allison, Eds. Baton Rouge, LA, USA: Louisiana State University, 2015, pp. 359–364. [Online]. Available: http://www.nime.org/proceedings/2015/nime2015_242.pdf

[12] B. Swift, "Chasing a feeling: Experience in computer supported jamming," in *Music and Human-Computer Interaction*, ser. Springer Series on Cultural Computing, S. Holland, K. Wilkie, P. Mulholland, and A. Seago, Eds. London, UK: Springer, 2013, pp. 85–99.

[13] R. Hamilton, J. Smith, and G. Wang, "Social composition: Musical data systems for expressive mobile music," *Leonardo Music Journal*, vol. 21, pp. 57–64, December 2011. DOI:10.1162/LMJ_a_00062

[14] G. Wang, "Game design for expressive mobile music," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, vol. 16. Brisbane, Australia: Queensland Conservatorium Griffith University, 2016, pp. 182–187. [Online]. Available: http://www.nime.org/proceedings/2016/nime2016_paper0038.pdf

[15] R. K. Sawyer, "Group creativity: Musical performance and collaboration," *Psychology of Music*, vol. 34, no. 2, pp. 148–165, 2006. DOI:10.1177/0305735606061850

[16] G. Schiemer and M. Havryliv, "Pocket Gamelan: Swinging phones and ad-hoc standards," in *Proceedings of the 4th International Mobile Music Workshop*, Amsterdam, May 2007.

[17] S. Greenberg and M. Roseman, "Using a room metaphor to ease transitions in groupware," Department of Computer Science, University of Calgary, Tech. Rep. 98/611/02, 1998.

[18] A. Carôt, P. Rebelo, and A. Renaud, "Networked music performance: State of the art," in *Audio Engineering Society 30th International Conference*, Mar 2007. [Online]. Available: http://www.aes.org/e-lib/browse.cfm?elib=13914

[19] A. Tanaka and P. Gemeinboeck, "Net_dérive: Conceiving and producing a locative media artwork," in *Mobile Technologies: From Telecommunications to Media*, G. Goggin and L. Hjorth, Eds. London, UK: Routledge, 2008.

[20] L. Gaye, R. Mazé, and L. E. Holmquist, "Sonic City: the urban environment as a musical interface," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, ser. NIME '03. Montreal, Canada: McGill University, 2003, pp. 109–115. [Online]. Available: http://www.nime.org/proceedings/2003/nime2003_109.pdf

[21] S. D. Chen, "A crude analysis of twitch plays pokemon," *arXiv preprint arXiv:1408.4925*, 2014. [Online]. Available: https://arxiv.org/abs/1408.4925

[22] P. Brinkmann, P. Kirn, R. Lawler, C. McCormick, M. Roth, and H.-C. Steiner, "Embedding Pure Data with libpd," in *Proceedings of the Pure Data Convention*. Weimar, Germany: Bauhaus-Universität Weimar, 2011. [Online]. Available: http://www.uni-weimar.de/medien/wiki/PDCON:Conference/Embedding_Pure_Data_with_libpd:_Design_and_Workflow

[23] R. I. Godøy, A. R. Jensenius, and K. Nymoen, "Chunking in music by coarticulation," *Acta Acustica united with Acustica*, vol. 96, no. 4, pp. 690–700, 2010. DOI:10.3813/AAA.918323

Does Singing a Low-Pitch Tone Make You Look Angrier?

Peter Ahrendt, Christian C. Bach
Aalborg University
Master Program in Sound and Music Computing
{pahren16; cbach12}@student.aau.dk

Sofia Dahl
Aalborg University Copenhagen
Department of Architecture, Design
and Media Technology
sof@create.aau.dk

ABSTRACT

While many studies have shown that auditory and visual information influence each other, the link between some intermodal associations are less clear. We here replicate and extend an earlier experiment with ratings of pictures of people singing high and low-pitched tones. To this aim, we video recorded 19 participants singing high and low pitches and combined these into picture pairs. In a two-alternative forced choice test, two groups of six assessors were then asked to view the 19 picture pairs and select the "friendlier", and "angrier" expression respectively. The result is that assessors chose the high-pitch picture when they were asked to rate "friendlier" expression. Asking about "angrier" expression resulted in choosing the low-pitch picture. A non significant positive correlation between sung pitch ranges from every participant to the number of chosen high-pitch resp. low-pitch pictures was found.

1. INTRODUCTION

Music and speech research often departs from the auditory signal, but we also know that human communication is much more than what meets the ear. For music, evidence have accumulated that support the view of music as multimodal, where other modalities interact with and affect the communication. This has for example been shown in studies on musical communication of expression [1, 2], emotion [3], structure [4], perception of sung interval size [5,6] and note duration [7].

Apart from musical contexts, auditory information is also contributing to our percept of a person's mood and facial expression. When listening to a speaker, we can hear if the person is smiling [8]. Ohala proposed that the origin of the smile (which involves teeth showing and ought to be perceived as threat) evolved as an acoustic rather than a visual signal [9]. In brief, Ohala suggested that the high-pitch resonances produced by stretching the lips in a smile helps to make an acoustic display of appeasement by appearing smaller and non threatening (see [10] for an overview).

Huron and colleagues investigated the intermodal association between pitch and emotion in two studies [11, 12]. In a first study, 44 participants were asked to sing neutral, high and low notes in a vowel of their own choice while their photograph was taken. Independent assessors were then asked to look at high and low pitch picture pairs and select "the more friendlier looking". The high pitch pictures were chosen significantly more often, also when the pictures were cropped to only show the eye-region [11]. A subsequent study [12] found the reverse association in the measured vocal pitch height (f0) for speakers reading different vowels with eyebrows raised or lowered. The effect size was small, but significant.

While these two studies mentioned above show the intermodal associations from two perspectives, the first study [11] did not provide any data of the actual produced pitches. It would be reasonable to assume that the range of pitches produced by a singer would affect the changes in facial expression and thereby also an assessors selection. Moreover, ratings for the "opposite" (negative) expression Anger would bring further support to the intermodal connection vocal pitch and facial expression.

The objective of this study was twofold: 1) To replicate the experiment done in [11] but with additional assessment of "angrier" expressions. 2) To investigate whether the actual produced pitches influenced the choices of assessors. To this end we asked participants to produce three pitches while being recorded on video. Images and produced pitches were extracted from the videos and later two groups of assessors selected the "angrier" or "friendlier" looking face in a forced choice test.

We hypothesize that faces singing low-pitched tones will be selected as "angrier" looking to a higher degree than high-pitch singing faces, while faces singing high-pitched tones will be selected as "friendlier" more often than low-pitch faces. Furthermore we hypothesize that the relative distance between high and low notes will influence the expressions of singers, so that pitch-pair pictures for comparably large pitch ranges will be more often identified according to the first hypothesis than faces of singers producing smaller pitch ranges.

2. DATA COLLECTION

2.1 Participants

For the picture retrieval part, 19 test participants were recruited immediately after having completed an approxi-

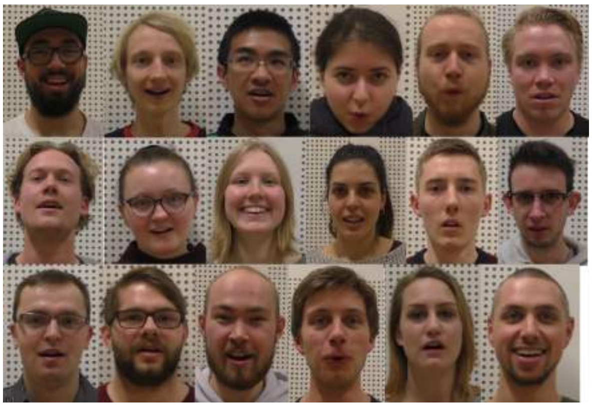


Figure 1. Some of the test persons participating in the test

mately 20 minutes long listening test. Participants were asked if they would like to be part of another experiment, involving them singing tones while recordings would be taken of them. While being a convenience sample, this way of recruiting also can be assumed to ensure that their emotional state would be relatively balanced as the test took place in a neutral testing environment. After being informed of the procedure, the volunteering participants gave their informed consent to being filmed. None of the participants received compensation for their participation.

2.2 Recording procedure

The environment was the Sound Lab of the Department of Architecture, Design and Media Technology, Aalborg University in Aalborg. The room has sound absorbing walls and no windows. The actual recordings took place in the main room which was divided by a wall with a window from an entrance room.

The participants were asked to stand in front of a regular camcorder placed on a tripod, and instructed to first produce a neutral effortless pitch, and then either a high followed by a low, or the reverse. High and low were in relation to their first pitch and the high-low order was determined using controlled randomization. The participants were instructed by variations of the following script presented by the experimenter:

We would like you to sing or produce a neutral tone which you are most comfortable with. After that tone please sing or produce a higher (lower) tone based on the neutral tone. After that the lower (higher) tone. You can sing whatever vowel you like. Please hold the tone a few seconds. Do you have any questions? [start recording] Now please sing the neutral tone. Good. Now the higher (lower) tone. Good. Now the lower (higher) tone. Good. [end recording]

To avoid that the situation would become stressful, which could have an influence on the facial expressions, we kept the task very simple. Thus, both vowel and pitches was left up to the participants (as done in [11]). Neither was any check for the actual vocal pitch range for each participant made.

While the participants were singing, the experimenter was standing behind the camera and focused on making the recording. That way participants would tend to look into the camera and not concentrate on the face of the experimenter. However, the participants were, not in all cases, asked to lock their eyes on the camera. Some of the participants wore glasses, which potentially could mask or interfere with the facial expression by covering, for example, the eyebrows. Well aware of the risk that less information would be conveyed, resulting in an increased difficulty for the assessors in the rating experiment, we did not ask the glasses to be removed.

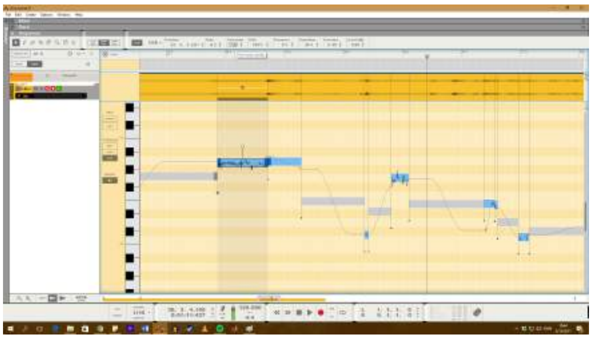


Figure 2. Tracking the pitch from the audio channel of the recorded video using the software Propellerhead Reason 9

The audio of the recorded video was exported into a DAW software called Reason 9 [13] to track and note the absolute pitches they produced in cents, MIDI notes and actual notes. In some cases the test participants were not able to sustain one clear note, so the average or the most prevalent pitch during each recording were chosen. The corresponding picture was then extracted from the video footage based on the time the pitch was produced, which would range from the first onset of the pitch produced (high or low) to the end. We did not settle for a fixed frame number across participants as they sometimes would blink, move around or in other ways reduce the clarity of the picture. Participants held, in most cases, tones three to four seconds long. In any case the picture chosen is one directly related to the pitch produced. The person who selected the pictures was not blind to the studies aim.

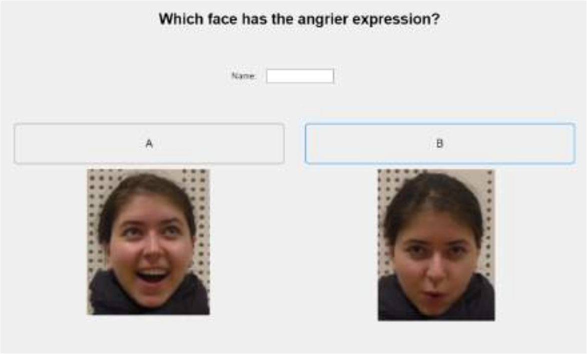
3. RATING EXPERIMENT

3.1 Participants

Two groups of six people from the local university campus volunteered as assessors to act in the rating experiment. The assessors were students who were currently in the building and approached by the experimenters. The rating of the pictures was made on a laptop on the site of the recruitment (usually open group working areas). None of the assessors had been involved in the first part of the test. The assessors received no compensation.

3.2 Stimuli and procedure

The produced images from the videos were assembled into pairs showing a high and low note per participant. A custom made testing interface made in the Matlab app designer [14] presented the picture pairs in a random order. For each picture pair, the program prompted the assessor to



Which face has the angrier expression?

Name:

A

B

Figure 3. Test interface for choosing the face with the "angrier" expression

select either the more "angrier/friendlier" expression (one for each of the two groups of raters respectively) in a two-alternative forced choice design (see Figure 3). After a decision was made, the next picture pair of high/low-note singing faces appeared and a new choice was prompted. The assessors were not able to revise there choice. The experiment took about three minutes.

4. RESULTS

4.1 Sung pitches

Figure 4 shows a boxplot for the sung high-pitch and low-pitch tones (calculated in MIDI notes). The red lines inside the box represent the median values of the midi notes whereas the lower edges of the boxes depict the first quartile (25 % of the data). The upper edges represent the location of the third quartile (75 % of the data). The data enclosed in the boxes shows 50 % of the whole data sets. The whiskers of the boxplot illustrate in our case outliers which are below of the first quartile or above the third one. Additionally gray lines were plotted which indicate individual tone pairs of each participant. The lines start from the corresponding midi note of the high-pitch tone and end at the corresponding midi note of the low-pitch tone from the same participant. This shall give an overview of the pitch ranges produced during the experiment. The pitch ranges differ partially heavily among the participants. One participant for example sang a high pitch of approx. 3900 cents (108 in MIDI notes) (corresponding to C4) and a low pitch of approx. 900 cents (78 in MIDI notes) (corresponding to F#1), whereas another participant sang a high pitch note of approx. 1300 cents (82 in MIDI notes) (corresponding to A#1) and a low pitch of approx. 1200 cents (81 in MIDI notes) (corresponding to A1). A typical range was sung for example from approx. 2500 cents (94 in MIDI notes) (corresponding to A#1) to approx. 1000 cents (79 in MIDI notes) (corresponding to G1).

As can be seen, the ranges partially overlap, which is to be expected given the different vocal ranges of males and females. 14 of the participants were males, five females. We made no difference in analyzing data from male and female participants. The mean low pitch sung was at 1339 cents (82 in MIDI notes) (corresponding to A#1) and the mean high pitch at 2490 (corresponding to A#2) (94 in MIDI notes). The overall range of the recorded pitches spanned 3244 cents between the lowest (corresponding to D#1) and highest (C4) pitches recorded, with an average of 1151 cents, which is slightly below an octave.

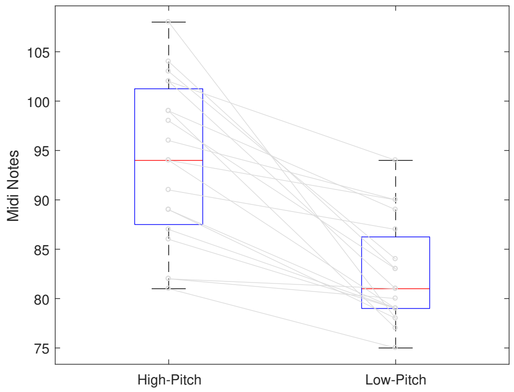


Figure 4. Boxplot for sung high- and low-pitch tones displayed in MIDI notes with individual tone pairs for each participant depicted with gray lines

	As.1	As. 2	As. 3	As. 4	As. 5	As. 6
As. 1	100					
As. 2	73.6	100				
As. 3	84.2	68.4	100			
As. 4	84.2	89.4	78.9	100		
As. 5	89.4	73.6	84.2	84.2	100	
As. 6	63.1	68.4	57.8	68.2	52.6	100

Table 1. "Angrier" inter-assessor agreement percentages

Please note that the depicted pitch ranges correspond to the pitch range the participants produced during the experiment. The participants were not tested in the maximum vocal pitch range they were able to produce.

4.2 Inter-Assessor Agreement

Table 1 and 2 show the inter-assessor agreement.

We calculated the inter-assessor agreement scores by summation of the agreeing choices between pairs of assessors, dividing it by the number of pictures. Hence, the scores do not indicate whether they chose the right picture or not, but whether two assessors selected the same picture. This was performed across all the pictures and assessor-pairs. If both assessors rated a given picture the same way, it counts towards the agreement aggregate.

Average assessor agreement was 74.6 % for selecting the face with an "angrier" expression. Assessor six seemed to differ from the others with an average of 62 %, at least 12 % below the rest. If assessor six is left out, the average agreement increases to 81 %. Table 2 shows the assessor agreements for selecting the "friendlier" face. For this group, there appears to be a higher agreement, with average scores between 73.64 % and 82.08 %, with a total average of 77.15 %.

	As.7	As. 8	As. 9	As. 10	As. 11	As. 12
As. 7	100					
As. 8	68.4	100				
As. 9	73.6	73.6	100			
As. 10	84.2	63.1	78.9	100		
As. 11	73.6	84.2	78.9	68.4	100	
As. 12	78.9	78.9	84.2	84.2	84.2	100

Table 2. “Friendlier” inter-assessor agreement percentages

Face	χ^2	df	p-value	High-Pitch	Low-Pitch
Angrier	21.93	1	$p < 0.01$	32	82
Friendlier	31.58	1	$p < 0.01$	87	27

Table 3. Associations between expression and number of high-pitch and low-pitch choices

Satisfied with the over all inter-assessor agreements for the two groups of assessors, we proceeded with the rest of the analysis and hypothesis testing.

4.3 Assessment of expression

The results of the choices of the assessors for the two experiments “angrier” and “friendlier” are summarized in table 3. As can be seen, the group assessing “angrier” predominantly selected the low-pitch faces (82 choices) compared to the high pitch faces (32 choices). Conversely, the other group of raters predominantly chose high-pitch faces as “friendlier” (87 choices) compared to low-pitch faces (27 choices).

Two Chi-square tests [15] confirmed that the sung pitches had a significant effect on the selected number of “angrier” and “friendlier” faces from the two groups of assessors. The χ^2 statistics from the test (calculated using the statistical software R) are also seen in Table 3.

The assessors’ picture selections support the hypothesis that facial expressions will give different emotional associations depending on the pitch sung. We now turn to the question whether a comparably large distance between high and low sung pitches would influence the facial expressions, thereby affecting the choises of assessors.

4.4 Correlation between Pitch range and Selections

To investigate whether there was a relationship between the difference between sung pitches and assessors’ choices, we calculated the produced *pitch range* in cents and compared this with the number of chosen high-pitch pictures for the “friendlier” facial expression and low-pitch pictures for “angrier” one respectively. The pitch range was calculated from the lowest to the highest pitch a participant sang. The relationship between the pitch range and the number of low-pitch faces chosen can be seen in Figure 5, while Figure 6 shows the relationship with number of high-pitch faces selected. As can be seen by the plotted trends there appear to be a weak positive relationship so that with an increase of pitch range the number of chosen pictures increases.

Face	r	t	df	p-value
Angrier	0.39	1.73	17	0.10
Friendlier	0.34	1.48	17	0.16

Table 4. Pearson’s product-moment correlation for number of chosen low-pitch (“angrier”) or high-pitch (“friendlier”) pictures and pitch range

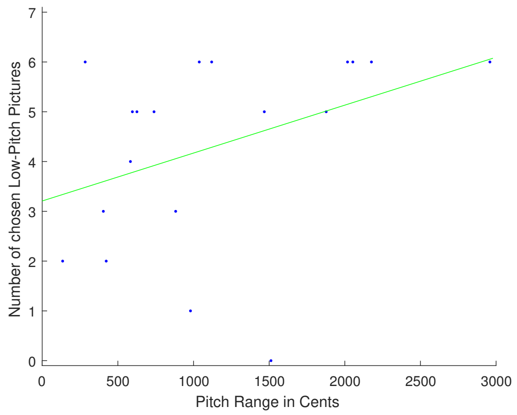


Figure 5. Correlation between the pitch range and number of low-pitch faces chosen. A linear regression curve is plotted in green.

A distribution investigation (Shapiro-Wilk normality test) resulted in the conclusion that the data for chosen number of pictures in each case was normally distributed while data for pitch range was not.

The statistics for two Pearson’s product-moment correlation [16] tests are presented in Table 4. The analysis showed no significant relationship between the range of pitches participants sang and the number of chosen high-pitch and low-pitch faces, respectively. Correlation coefficients were 0.39 and 0.34 for the selected low and high notes pictures respectively.

An outlier with a pitch range of 1512 cents can be seen in both Figures 5 and 6 where the number of chosen pictures for either high-pitch pictures or low-pitch pictures is zero. This means that two different groups of assessors rated the same pair of pictures of the same participant completely the opposite of what was predicted in our hypothesis. Notably the pitch range produced by this singer was near the average pitch range.

5. DISCUSSION

The results from our rating experiment showed that the high-pitched faces were selected as more friendlier than the low-pitch faces, replicating the results in [11]. In addition, our results also show the reverse relationship, with low-pitched faces selected as more angrier in comparison with the high-pitched faces. Our findings bring further support to intermodal associations between facial expression and vocal pitch height.

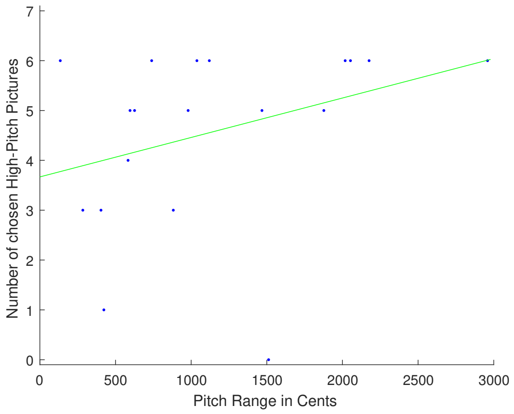


Figure 6. Correlation between the pitch range and number of high-pitch faces chosen. A linear regression curve is plotted in green.

A possible explanation for such intermodal associations could be the usefulness of conveying different perceived size (and hence threat) to other individuals (see e.g. [10, 12, 17]). In every day communication we typically shift between appeasing and threatening displays. For instance, infant-directed speech typically presents a non-threatening display with high pitch and raised eyebrows [18] while the infants older siebling might get told off by the same parent now frowning and using lower pitch in a more threatening display. Since parents come in many sizes with different vocal pitch ranges, it seems reasonable to assume that a combination of visual (raised or lower eyebrows) and vocal (high or low pitch) information would make the display less ambiguous to the receiver.

The reason for the eyebrow position in giving a friendly, non threatening, or an angry, more threatening display is subject to speculation. In a commentary to Huron et al. Ohala [17] suggested that raised eyebrows give the impression of proportionally larger eyes – something characteristic for newborns and infants. Since small children have relatively large eyes and are non threatening, giving the impression of larger eyes would also give the association of being smaller, less mature and non threatening. Conversely, the impression of smaller eyes would be associated with being more mature (and so possibly more of a threat) [17].

We hypothesized that the difference in pitch height, rather than the actual pitches per se, would influence the strength of the communication to the assessors. However the actual range in pitches produced by the singers in our study appeared to be of less importance, disproving our second hypothesis. The Pearson correlation between produced pitch range (in cents) and pitch-pictures selected by assessors, was 0.39 and 0.34 for the selected low and high notes pictures respectively. Although fairly small and non significant, the positive correlation is in line with the results of Thompson and Russo [5], who reported high correlation for observers rating sung interval size based on visual video information only. Thompson and Russo also measured the change in position of markers on the head, mouth, and eyebrows of the trained singers and found a positive correlation with sung interval size. Despite non-uniform data and using non-expert singers as participants in our study we find indication of similar relationship.

Some methodological choices in our study deserve further discussion. Firstly, the task of the singing participants was kept simple and both the vowel and pitches to sing was left to the individual (similar to the original study [11]). Since participants were not explicitly asked to maintain the first vowel they chose when singing the second note, three out of 19 participants shifted to a different vowel for the second note (see e.g. Figure 3). Different vowels affect the facial expressions and it is possible that this influenced the assessors choices. However, the same "free" instructions were used in the study by Huron et al. who found a significant effect also when only the eye-region was shown to assessors [11].

If the facial expressions associated with low and high pitches are part of a more general intermodal association, choosing different vowel could also be part of the signalling system. Ohala proposed that smiling helps to shape the acoustic signal to give a "friendlier", more appeasing impression by shaping the filter to boost the higher frequencies [9]. As noted by Huron "the 'opposite' of the smile may not be the frown, but the pout" [10], something that has been observed as threat signal in many animal species. Pouting would indicate the production of rounded vowels that help to lengthen the vocal tract and produce lower resonances. In our experiment we counted nine participants who were pouting. Most of the pouting pictures were for the low pitch but in two cases participants also pouted with the high pitch, one of these participants produced relatively low pitches in general.

All singers started by producing a neutral, comfortable, tone, and in relation to this tone the higher and lower pitches were freely selected by the singers. As can be seen in Figure 4, the actual produced pitch ranges differed considerably. Some singers produced large differences while others chose a more modest range. If the facial expressions are related to the high and low pitches relative to each person's individual vocal pitch span, an assessment of this one would have to be made. One way to address this in future studies could be to let a professional voice expert blindly assess how strained the voice appears for the produced notes.

A curious case is the specific set of picture pairs from one of the singers in our study that appeared as an outlier in Figures 5 and 6. Although the singer (Danish, male, had little facial hair, no glasses) produced notes with a reasonable interval range, the assessors consistently chose the low-note picture for the "friendlier" selection and the high-note picture for the "angrier". Both pictures show the participant smiling with his head slightly turned, but the smile in the high-pitch picture may be described as more of a smirk while the singer is looking in a different direction. The low-pitch picture, on the other hand shows the participant looking straight into the camera. This could be a possible explanation that the low-pitch face appeared to present someone more approachable and friendly.

Apart from eyebrow height, there are other facial cues that observers use to judge what type of behaviour or personality an individual is likely to have. For instance, different facial morphology in both humans and chimpanzees reveal personality traits to observers [19] and an association between the width-to-height ratio and aggression ratings of static images showing neutral faces has been reported [20]. In our study we showed within-participant pairs of pictures to be compared by the assessors, but still some of these faces could have provided strong static cues not associated with the task of singing high and low notes. By rating the aggressiveness or friendlieness of individual pictures, possibly combined with eye tracking, it would be possible to better identify the specific role of the eyebrows in the assessment of the faces.

6. CONCLUSION

When singing notes of different pitch height, untrained singers' facial expressions are altered in such a way that when assessors are asked to select the "friendlier" face, they would choose the high-pitch picture to a higher degree over the low-pitch picture [11]. We could replicate this result from our experiment. Furthermore, we could also show the opposite relation where assessors instructed to select the "angrier" face, to a higher degree choose the low-tone picture over the high-pitch one. Hence, a task normally associated with music appear to affect dynamic facial expressions in a way that observers detect as emotional expressions influencing their choices of "angrier" and "friendlier" above level of chance.

Acknowledgments

The authors would like to thank all singing participants and assessors.
Authors Ahrendt and Bach were responsible for the main part of the work and set up the experiment, performed data collection and analysis under the supervision of author Dahl as part of a course in the Sound and Music Computing master program at Aalborg University. All authors participated equally in the writing of the final manuscript.

7. REFERENCES

[1] J. W. Davidson, "Visual perception of performance manner in the movements of solo musicians," *Psychology of music*, vol. 21, no. 2, pp. 103–113, 1993.

[2] M. Broughton and C. Stevens, "Music, movement and marimba: An investigation of the role of movement and gesture in communicating musical expression to an audience," *Psychology of Music*, vol. 37, no. 2, pp. 137–153, 2008.

[3] S. Dahl and A. Friberg, "Visual perception of expressiveness in musicians' body movements," *Music Perception: An Interdisciplinary Journal*, vol. 24, no. 5, pp. 433–454, 2007.

[4] B. W. Vines, C. L. Krumhansl, M. M. Wanderley, and D. J. Levitin, "Cross-modal interactions in the perception of musical performance," *Cognition*, vol. 101, no. 1, pp. 80–113, 2006.

[5] W. F. Thompson and F. A. Russo, "Facing the music," *Psychological Science*, vol. 18, no. 9, pp. 756–757, 2007.

[6] W. F. Thompson, F. A. Russo, and S. R. Livingstone, "Facial expressions of singers influence perceived pitch relations," *Psychonomic bulletin & review*, vol. 17, no. 3, pp. 317–322, 2010.

[7] M. Schutz and S. Lipscomb, "Hearing gestures, seeing music: Vision influences perceived tone duration," *Perception*, vol. 36, no. 6, pp. 888–897, 2007.

[8] V. C. Tartter and D. Braun, "Hearing smiles and frowns in normal and whisper registers," *The Journal of the Acoustical Society of America*, vol. 96, no. 4, pp. 2101–2107, 1994.

[9] J. J. Ohala, "The acoustic origin of the smile," *The Journal of the Acoustical Society of America*, vol. 68, no. 51, pp. 533–533, 1980.

[10] D. Huron, "Affect induction through musical sounds: an ethological perspective," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 370, no. 1664, p. 20140098, 2015.

[11] D. Huron, S. Dahl, and R. Johnson, "Facial Expression and Vocal Pitch Height: Evidence of an Intermodal Association," *Empirical Musicology Review*, vol. 4, no. 3, pp. 93–100, 2009.

[12] D. Huron and D. Shanahan, "Eyebrow movements and vocal pitch height: evidence consistent with an ethological signal." *The Journal of the Acoustical Society of America*, vol. 133, no. 5, pp. 2947–52, 2013.

[13] Propellerhead Software AB, "Reason 9 Operation Manual," Tech. Rep., 2011.

[14] "Matlab Appdesigner," 2017. [Online]. Available: https://se.mathworks.com/products/matlab/app-designer.html

[15] A. Agresti, *An Introduction to Categorical Data Analysis*, 2nd ed. Gainesville: John Wiley & Sons Inc., 2007, vol. 22, no. 1.

[16] D. J. Best and D. E. Roberts, "The Upper Tail Probabilities of Spearman's Rho," *Wiley for the Royal Statistical Society*, vol. 24, no. 3, pp. 377–379, 1975.

[17] J. Ohala, "Signaling with the Eyebrows–Commentary on Huron, Dahl, and Johnson," *Empirical Musicology Review*, vol. 4, no. 3, pp. 101–102, 2009.

[18] S. Chong, J. F. Werker, J. A. Russell, and J. M. Carroll, "Three facial expressions mothers direct to their infants," *Infant and Child Development*, vol. 12, no. 3, pp. 211–232, 2003.

[19] R. S. Kramer, J. E. King, and R. Ward, “Identifying personality from the static, nonexpressive face in humans and chimpanzees: evidence of a shared system for signaling personality,” *Evolution and Human Behavior*, vol. 32, no. 3, pp. 179–185, 2011.

[20] S. N. Geniole, A. E. Keyes, C. J. Mondloch, J. M. Carré, and C. M. McCormick, “Facing aggression: Cues differ for female versus male faces,” *PLOS one*, vol. 7, no. 1, p. e30366, 2012.

Monitoring and Supporting Engagement in Skilled Tasks: From Creative Musical Activity to Psychological Wellbeing

Juan Carlos Vasquez, Koray Tahiroğlu, Niklas Pöllönen
Department of Media
Aalto University
School of ARTS
FI-00076 AALTO Finland
firstname.lastname@aalto.fi

Johan Kildal
IK4-TEKNIKER
Iñaki Goenaga 5
Eibar 20600
Spain
johan.kildal@tekniker.es

Teemu Ahmaniemi
Nokia Technologies
Espoo, Finland
teemu.ahmaniemi@nokia.com

ABSTRACT

While good physical health receives more attention, psychological wellbeing is an essential component of a happy existence. An everyday source of psychological wellbeing is the voluntary practice of skilled activities one is good at. Taking musical creation as one such skilled activity, in this work we employ an interaction method to monitor varying levels of engagement of musicians improvising on a desktop robotic musical interface (a network of intelligent sonic agents). The system observes the performer and estimates her/his changing level of engagement during the performance, while learning the musical discourse. When engagement levels drop, the musical instrument makes subtle interventions, coherent with the compositional process, until the performer's engagement levels recover. In a user study, we observed and measured the behaviour of our system as it deals with losses of performer focus provoked by the controlled introduction of external distractors. We also observed that being engaged in our musical creative activity contributed positively to participants' psychological wellbeing. This approach can be extended to other human activities.

1. INTRODUCTION

Through life, individuals develop skills in activities that they find interesting and at which they excel. Such skilled activities often combine intellectual, crafting and creative dimensions: from cooking and knitting to writing, drawing or, in this case, performing with a new musical interface. Practising such activities in an autonomous and voluntary way can bring great satisfaction and sense of purpose to a person, who finds the opportunity to challenge the limits of one's own acquired mastery. The confluence of these three characteristics in an occupation - autonomy, mastery and purpose - has been identified and persuasively proposed as a popular recipe for personal fulfilment at work [1]. In the wider scope of life, these three characteristics are also part of a set of six ingredients that have been identified as important for a healthy, actively-positive psychology in adult life: autonomy, (environmental) mastery, purpose (in life), self acceptance, positive relations with other people and personal growth [2]. Psychological wellbeing is an essential aspect of general wellbeing that traditional medicine does not take care of as a priority in every case. However, it is essential for a happy existence, and being engaged in skilled tasks provides much of such satisfaction and subsequent psychological wellbeing [3].

In the work that we present here, we focus on exposing the centrality of the individual's engaging experiences in psychological-wellbeing enhancing musical activities. We seek to amplify the experience by mediating between the person and the activity in two complementary ways: (i) by monitoring the level of engagement of the person during a musical activity, and (ii) by having the object of the activity itself respond in subtle but meaningful ways to the music composition led by the person, in order to maintain consistently high levels of engagement. We build on work that we developed earlier [4], regarding the machine recognition and monitoring of the engagement level of an individual that is immersed in a musical task with digital environments. The work so far has focused on recognising levels of immersion of musical performers during creative improvisation with a desktop Network Of Intelligent Sonic Agents (NOISA), the musical instrument itself. By combining the monitoring of facial expressions, bodily movements, actions of the hands and the evolution of the sonic production, an initial setup has been validated to successfully estimate levels of engagement at each time of the creative musical process [5].

As the main technical contribution of this paper, we add to the engagement monitoring functionality that NOISA had, a *new response module*, which preserves high levels of engagement in a musical activity when these appear to fade. In this paper, we report a new study conducted in the new NOISA environment that aimed at evaluating the functionality of the response module and the music composition task's effect on the participant's psychological wellbeing. First, we evaluated the system's functionality to identify instances in which we observe a drop in the level of engagement of the person with the musical activity. Secondly, we evaluated the response module's functionality to intervene in subtle ways in order to "rescue" dropping levels of engagement back to higher levels that are associated with a satisfactory creative experience. Lastly, as happiness and wellbeing crucially depend on the activities we engage in [3], in this user study we aimed at understanding whether a satisfactory musical experience could indeed contribute towards the preservation of psychological wellbeing, as we proposed in the beginning.

2. RELATED WORK

In spite of being a commonly mentioned parameter in music research, the mere act of obtaining a unified and consistent definition for 'engagement' represents a surprising challenge. In the field of education, Appleton [6] points the discussion towards separating engagement from motivation: the latter [7] provides an answer to the reasoning behind a specific behaviour depending on the characteristics of the personal energy invested in said situation, whereas engagement refers to the level of focused involvement [8], or "energy in action, the connection between person and activity" [9]. When applied to musical activity, engagement may also have a positive connotation, described as a state of sustained, undivided attention and emotional connection towards creative musical performance [10] or sometimes as a neutral parameter prone to be measured and evaluated in interdependent values for attention, emotion and action [11].

Similarly, there are other studies exploring music as a creative activity that contributes to the general wellbeing. An example demonstrates how active music-making improves wellbeing and overall comfort in a large sample of hospitalised children with cancer [12]. This study draws upon previous research featuring live music therapy that promotes interaction through voice and body expressions, recognising the benefits of interactivity in opposition of listening passively to music as a therapy [13]. While previous studies delineate a methodology for therapy rather than the design of an interactive system, there are examples of interactive tools aimed at positively impacting in wellbeing through rhythmical movement with music as a central element [14, 15]. There is also the precedent of an interactive application to engage patients with dementia in active music making without pre-existing skills [16]. In the same vein, we found an interactive technology tested in children with disabilities and aimed to study how music interaction can "become potentially health promoting" [17].

As a concept, psychological wellbeing is subjective, relative and prone to change with ease. However, it is possible to find specific sonic characteristics that have proven to affect positively an individual's perception of comfort and wellbeing. For instance, in the context of music, some studies indicate that fast and slow tempos are associated with happiness and sadness, respectively, as are major and minor modes [18, 19]. Loud music has proven to be in cross-cultural studies as an universal cue for anger [20]. Timbre is ambivalent, although several studies indicate that attenuation of high frequencies can be a difference between anger (unfiltered) and tenderness (attenuation) [19]. Although affective associations of both tempo and mode are fairly well established, effects of other musical characteristics are poorly understood. On the cognitive level, David Huron's ITPRA (Imagination-Tension-Prediction-Response-Appraisal) theory studies the relationships between affective consequences and expectancies in music [21], establishing clear connections. Evidence also suggests that some auditory illusions, such as Binaural Auditory Beats, are associated with less negative mood, notably affecting performance in vigilance tasks of sustained concentration, particularly when they occur in the beta range (16 and 24 hz) [22].

Regarding the concept of musical interaction being perceived as rewarding, the Orff approach states that improvisation with pentatonic scales, with bars and notes removed, is always satisfactory and encourages freedom from the fear of making mistakes [23]. The goal is for everybody to experience success and appreciate the aesthetic in music almost immediately, rather than having to learn notes and rhythms before anything satisfying is accomplished. Percussive patterns were important in his theory, as they appeal to a primal essence in human beings and allow tempo and beat to be experienced rather than learned. We incorporated many of these ideas into our work while designing the response module and its implementation in a music-making creative activity that can link cognitive features and physical movements together to pay attention for immediate responses. This is a generic hypothesis of research in cognitive neuroscience of music and the concept of music, mind and body couplings that our project takes into account [2, 24]. The former points out the benefits of music as a therapeutic tool and it can improve recovery in the domains of verbal memory and focused attention. The latter indicates its impact on social and individual development through active engagement with music.

3. NOISA

As mentioned, NOISA (Network Of Intelligent Sonic Agents) is the robotic musical interface that we took into use and built on for this research work. NOISA consists of three instruments for music creation, a Microsoft Kinect 2 camera and a central computer. The three instruments are physically identical from each other. Each instrument is built inside a plastic box. The interface for playing each instrument consists of two handles designed to provide an ergonomic form for grasping them from several directions, which can also position themselves actively, thanks to DC motors (the robotic interactions with the physical medium). The handles are attached to a motorised fader, which served both as position indicator and as active position controller. The hardware in each box includes an Arduino, a Raspberry Pi, two motorised faders with extra large motors, two capacitive touch sensors on the handles, and an embedded speaker. In each instrument, sound production is largely based on granular synthesis and phase vocoder algorithms de-fragmenting radically small fragments of pre-existing pieces from the classical music repertoire.

3.1 Response Module

Based on Huron's ITPRA theory, we concluded that working in a stronger Prediction-Response musical connection would likely produce an affective response evaluated as positive [21]. Huron also proposes that recognisable patterns are a common characteristic in music associated to positive emotional responses. Thus, we decided that for the current iteration of NOISA, the system would recognise, store and playback musical patterns of 2 seconds of activity, which the performer could easily identify based on the data inputted. In order to avoid to solely reproduce an exact copy of the gesture performed by the performer, we also programmed a set of variations, based on the music composition technique of motivic-through composition, also known as thematic development. This technique has been used successfully in classical music from the common practice period (barroque to late XIX century), and also in post-millennial music, including mainstream genres [25].

We defined the gesture by recording a maximum of seven-second long movements. The data recorded included the engagement value, device identification, energy (measured in decibels), spectral centroid (average of most prominent frequency ranges over time), movement of the handlers and time-stamps. Recording was initiated when a new movement was detected, and stopped when inactive for longer than one second. The gesture playback was activated depending on the performer's engagement in the task, activating more often when the engagement was low. The playback gestures were then chosen from a fifteen-minute interval. Gesture data was measured against weights that were defined by the energy mean value, spectral centroid, spectral energy and amount of time elapsed since the recording was made. The result was that, when the engagement was higher, a larger set of chosen gestures was used. When lower, newer patterns were chosen matching the readings from the performers currently playing, to make the pattern more recognisable.

The first choice of instrument for playback (out of the three autonomous sonic agents) was the instrument on which the gesture to be reproduced had been recorded. If that instrument was in use at that time, one of the other two instruments was randomly selected instead. There were four variations of response playback: the recording itself, an inverse (negative), a reverse and an inversed-reversed version of the recording. The inverse was mapped into the right range with the help of the highest and the lowest values. The engagement level of the performer was the deciding factor for the chosen gesture variation. When less engaged, the pure recording was the most common choice. When engagement became higher, the performer experienced a larger set of variations.

3.2 Hardware Design

The new response module required more precision than the initial NOISA design could handle, so a new inner structure for the instrument was designed. Most of the hardware parts were custom made with 3D printing, computer numerical control (CNC) milling and laser cutting (see Figure 1). After a few iterations we reached the current state. Aluminium tubes were added to make the handles stronger yet light. The slide-bearings for the aluminium tubes were CNC-milled for an exact fit. The handles were redesigned to manipulate the added weight, using additional timing belts and aluminium pulleys. The benefit of these changes was lower friction from the handles. For the outer shell to be strong enough to hold the new structure, we added supporting layers and attached everything together with custom-made corner-irons. The handles of the instrument were also redesigned, with holes for the aluminium rods and added space for the pressure sensors. To detect the pressure, two pressure-sensitive conductive sheets were sandwiched between layers of copper tape. By reading the change of the resistance, the amount of pressure could be detected.

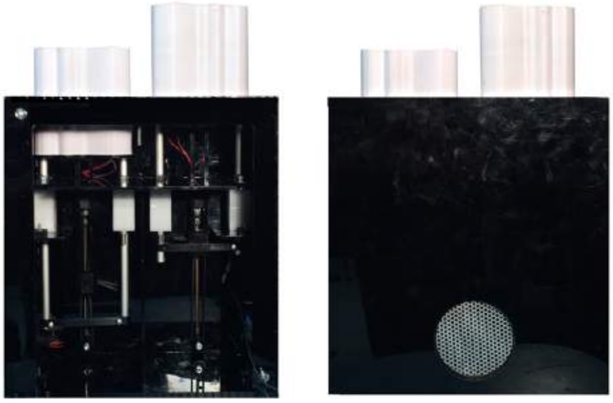


Figure 1. New interface design of the NOISA instrument

4. USER STUDY

To review our main research goals in this work, we wanted to observe, first, if participating in the skilled activity of improvising music with our accompanying robotic instrument showed any signs of being subjectively beneficial for people. Such outcome would suggest that our improvisation activity might be eligible as one of the many possible skilled tasks in the scope of this study (see again our discussion about this in the introduction). Second, we wanted to observe our interactive system in action, both monitoring the levels of engagement with the activity (a functionality already validated for earlier versions of this instrument), and, more crucially, intervening (i.e., coaching) when the performer's level of engagement dropped to low levels. As the main technical contribution, we wanted to observe if the system was able to refocus the flow of the performer's activity until higher engagement levels were again measured.

To do all of that, we designed and conducted a two-part user study in which 15 voluntary participants (8 females) took part. The first part aimed at recording a subjective measure of wellbeing before and after a performance session in which a person could improvise under normal conditions. In the second part, in contrast, we provoked the loss of focus of the participant in the improvisation activ-

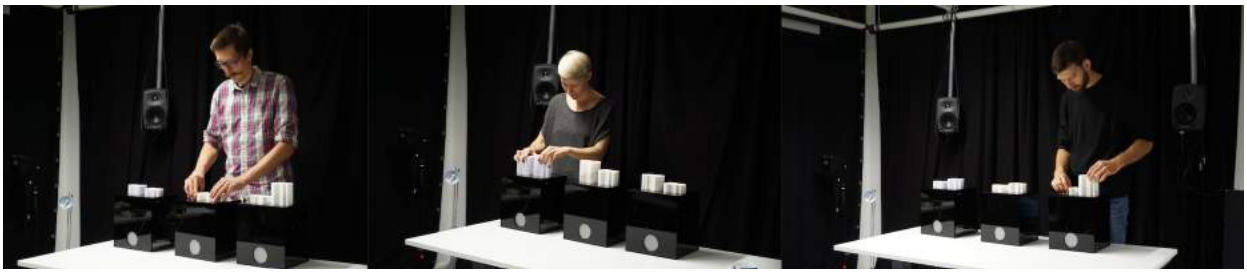


Figure 2. Users developing the task

ity, and we observed if the system could observe the corresponding drop in engagement, and if the instrument could intervene collaboratively until engagement was recovered.

After a general description of the whole session and after obtaining consent, the first part of the user study started with the participant filling out the PANAS questionnaire (The Positive and Negative Affect Schedule) [26]. This is a validated questionnaire that evaluates positive and negative affects using a psychometric scale, and the change in those within a specific period of time during which some treatment activity took place (in our case, a performance session with our collaborative musical instrument). PANAS uses 10 descriptors per category of affects, showing relations between them with personality stats and traits. PANAS has been successfully applied to measure overall psychological wellbeing before, and after a musically creative activity [27]

The participant was then introduced to the NOISA instrument, with explanations about how it works, followed by a period of hands-on familiarisation with it (typically for about three minutes). Once the participant felt confident enough, the first music performance task was described, which required performing a piece according to a somewhat restrictive set of rules (we wanted to reduce variability in the performing activities at this stage in the study). In particular, the participant would be asked to start performing with a specific instrument (Agent 1), then repeat the given gesture with the middle instrument (Agent 2), finishing with inputting the same repeated movement in the third instrument (Agent 3), resulting in a piece that lasted for between three and five minutes. Once this was done, NOISA would have typically started to make automatic responses. At this moment, the participant was asked to explore freely any of the agents, and once the participant was ready to finish, she/he had to manipulate the handles in Agent 1. The participant was required to stop any input activity as an indication of the end of the music task.

After finishing the task, the participant was asked to fill in again the PANAS Questionnaire in order to compare the impact of the task in the overall sense of psychological wellbeing. Following that, the participant was asked to evaluate the system's usability by filling in the online version of the System Usability Scale (SUS) designed by our research group to facilitate the data gathering and analysis. SUS has ”provided with that high-level measurement of subjective usability, and over the years the work done by other researchers has indicated that it is a valid and reliable tool” [28].

For the second part of the study (designed to observe the engagement tracking and recovery functionalities of our interactive system), we asked the participant to repeat the same musical task, extending it for a period of four minutes. This time we tracked and logged the level of engagement over time. We scheduled two events sufficiently distracting at 1:30 (door opening, person coming in and loudly manipulating a cardboard box) and a more sublte event at 3:00 (phone ringing). We drew inspiration from previous studies measuring the impact of distractions in psychological wellbeing during a skilled activity [29]. The participant was not aware that such events would take place, nor he/she was prepared for them. The total duration of the test was approximately 30 minutes (see Figure 2).

5. RESULTS

The PANAS scores are divided in Positive Affects (PA) and Negative Affects (NA). According to the PANAS documentation, the mean PA scores for momentary (immediate) evaluation are 29.7 (SD = 7.9). Scores can range from 10 to 50, with higher scores representing higher levels of positive affect. For Negative Affects the mean NA score for an equally momentary evaluation are 14.8 (SD = 5.4). Scores can range from 10 to 50, with lower scores representing lower levels of negative affects.

During our tests, we obtained a PA score mean (Before the activity) = 30.57 (SD = 3.89) and a PA score mean (After the activity) = 34.21 (SD = 5.14). In addition, we collected a NA Score (Before the activity) = 15.57 (SD = 6) and a NA Score (After the activity) = 12.30 (SD = 3.30). This means an overall improvement of 3.64 PA points and 3.27 NA points. By adding PA and NA scores, we found that the overall improvement in the reported psychological wellbeing was of 6,91 points, which in a range of 10 to 50 in the PANAS range represents an overall improvement of 17.27% after 3 minutes of activity with our system. In addition, 30% of the participants reported an improvement of more than 10 points in the PA affect score

In our paired before/after test, for Negative Affects the two-tailed p value equals 0.0029 (t = 3.5979), a statistically significant difference (significance level $>$ 99%). Regarding the Positive Affects, we obtained a p value of 0.0699 (t = 1.9750), which is also statistically significant (significance level $>$ 95%).

Regarding usability, the System Usability Scale research drew the average of 68 points in the scale (from 0 to 100). Sauro [30] found that when people rate a system or product with a SUS score of 82, they are likely to recommend a system or product to a friend or colleague (Net Promoter Score). After evaluating our system, the participants gave it a SUS mean value of 73.9 (SD = 11.33) which means 5.9 points above average. One third of the participants gave our system a SUS score of 82 or higher, potentially becoming a "promoter".

Finally, our engagement graphs over time showed that the system was able to identify and measure a distraction event in 73.33% of the participants, impacting the overall engagement by 0.2 points or more, in a scale of 0 (total disengaged) to 1 (full engagement). Our engagement scale is also divided into categories, where 0 to 0.3 means low engagement, 0.3 to 0.6 medium engagement and 0.6 to 1 high engagement. Calculating every time a distraction had an impact in the engagement, there was an overall average of 0.41 sudden decrease (SD = 1.40). As the NOISA system reacts once it detects a drop in the overall engagement, the level of focus was effectively regained to a stable level before the distraction event in a mean time of 10.07 seconds (SD = 7.15). Two engagement over time graphs for the participants 10 and 11 (see Figure 3 and 4) are included to demonstrate an example of a case where our engagement prediction system detected both distractions (90 and 180 second mark, respectively), managing to regain focus of the participant through autonomous counteractions.

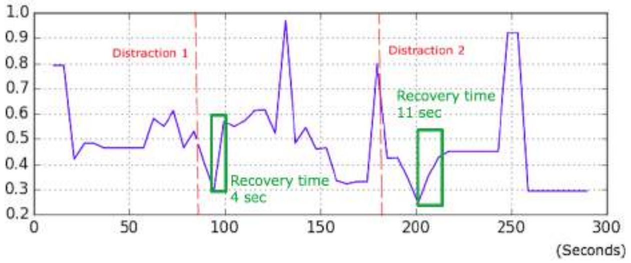


Figure 3. Engagement over time graph of the participant 10

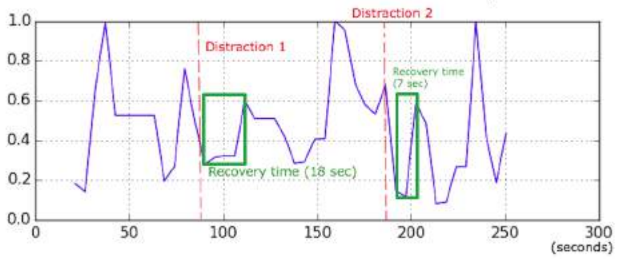


Figure 4. Engagement over time graph of the participant 11

6. ANALYSIS AND DISCUSSION

The results presented above show that the design of the response module provided beneficial results for the participants. The response module could successfully maintain and strengthen the level of engagement in a skilled activity of musical improvisation with NOISA instruments. The resulting experience of such interaction was also likely to have contributed positively to the psychological wellbeing of the participants.

Judging from the quantitative data obtained through the PANAS questionnaire, the average levels of Positive Affects were higher after finishing the test, and the Negative Effects were lower after completing the task. In case of Negative Affects, the difference was found to be very statistically significant. In the PANAS scale, the Negative Affects scores decreased from above the mean (15.57) to below it (12.30). Even though the impact on the Positive Affects was found to be not quite statistically significant, we consider the overall improvement of 17.27% positive considering the short amount of time spent with the system during our task. The creative activity facilitated by NOISA system proved to be statistically significant to impact on the Negative Affects described in PANAS (Distressed, Upset, Guilty, Hostile, Irritable, Ashamed, Nervous, Jittery and Afraid), which shows that the musical task showed some capabilities during the test that can be successfully applied to help reducing the subjective perception of worry, stress and other negative personality traits of the participants.

The engagement estimation results indicated effectiveness when detecting and maintaining the focus level of the participants. In that sense, being able to measure at least one of the distractions in 73.33% of the participants is satisfactory, considering that while being observed, the participant could feel compelled to keep his/her attention to the task at hand. This situation would avoid us to detect any distractions in their physical body movements. The response module was capable of recovering attention and focus from the participants in an average time of 10.07 seconds after the distractions. We can note here that our current contribution to NOISA system provided an engaging interaction that was gradually and progressively built up by the participant's active and continuous involvement in the musical activity with NOISA instruments.

Similarly, we observed that both our new hardware design and response module were evaluated by the participants as easy to use, obtaining consistent above-average scores in the SUS scale. In addition, a large portion (one third) of the participants cataloged it above the 82 mark, which is particularly positive considering that we asked non-specialist individuals to evaluate a new musical interface. Our sample consisting of 15 participants might be considered small; however, Tullis and Stetson [31] have already demonstrated the reliability of SUS in smaller samples (8-12 participants).

7. CONCLUSIONS

Based on previous literature, it is clear that designing for wellbeing is strongly linked with the psychological understanding of wellbeing, positive experience and physical activity [3]. The purpose of our project was to establish a situated musical activity for the user to be engaged in, attempting to achieve the goal of enhancing psychological wellbeing. Our design was grounde in a concrete context (creative musical overall process) and guided by the functionality (hand-held interface) as well as through subtle interaction (counter-responsive musical actions). We observed and measured the behaviour of our system as it deals with losses of performer focus provoked by the controlled introduction of external distractors. During the tests, we observed indications that being engaged in our musical activity contributed positively to participants psychological wellbeing. Encouraged by our initial results, we plan to continue our research by including in a future study a comparison with a control group, which will enable us to draw in a longer term more comprehensive conclusions of the impact of the activity with NOISA in the pyschological wellbeing of the user.

We also aim to investigate the application of a similar engagement-monitoring technique in a broader scope of skilled, potentially mentally absorbing actions with people working in different occupations. A foreseen psychological wellbeing application scenario that will be implemented in the future phases of the project, could be the non-intrusive recognition of changes in patterns of engagement that a person presents when carrying out different creative tasks with digital environments, which could signal changes in their mood, psychological health or deterioration of their cognitive capacity (early diagnostic goal). This application requires a new hand-held interface to be designed and implemented which will be fully integrated with the engaging monitoring and the new response module in NOISA system. Through subtle external actions, the new interface and the digital environment could assist the person to regain focus and engagement with the creative task in hand, which might be beneficial for the successful completion of the task (assistive living goal), and to do so with an enhanced sense of control, accomplishment and satisfaction (psychological wellbeing goal) in a creative process.

Acknowledgments

This project was developed thanks to the support of the Nokia Research Centre in Finland.

8. REFERENCES

[1] D. H. Pink, *Drive: The surprising truth about what motivates us.* Penguin, 2011.

[2] C. D. Ryff, "Psychological well-being in adult life," *Current directions in psychological science*, vol. 4, no. 4, pp. 99–104, 1995.

[3] S. Diefenbach, M. Hassenzahl, K. Eckoldt, L. Hartung, E. Lenz, and M. Laschke, "Designing for well-being: A case study of keeping small secrets," *The Journal of Positive Psychology*, pp. 1–8, 2016.

[4] K. Tahiroğlu, T. Svedström, and V. Wikström, "Noisa: A novel intelligent system facilitating smart interaction," in *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, ser. CHI EA '15. ACM, 2015, pp. 279–282. [Online]. Available: http://doi.acm.org/10.1145/2702613.2725446

[5] K. Tahiroglu, T. Svedström, and V. Wikström, "Musical engagement that is predicated on intentional activity of the performer with noisa instruments," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, E. Berdahl and J. Allison, Eds., May 31 – June 3 2015, pp. 132–135. [Online]. Available: http://www.nime.org/proceedings/2015/nime2015_121.pdf

[6] J. J. Appleton, S. L. Christenson, D. Kim, and A. L. Reschly, "Measuring cognitive and psychological engagement: Validation of the student engagement instrument," *Journal of School Psychology*, vol. 44, no. 5, pp. 427 – 445, 2006, motivation. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0022440506000379

[7] M. L. Maehr and H. A. Meyer, "Understanding motivation and schooling: Where we've been, where we are, and where we need to go," *Educational Psychology Review*, vol. 9, no. 4, pp. 371–409, 1997. [Online]. Available: http://dx.doi.org/10.1023/A%3A1024750807365

[8] J. M. Reeve, H. Jang, D. Carrell, S. Jeon, and J. Barch, "Enhancing students' engagement by increasing teachers' autonomy support," *Motivation and Emotion*, vol. 28, no. 2, pp. 147–169, 2004. [Online]. Available: http://dx.doi.org/10.1023/B%3AMOEM.0000032312.95499.6f

[9] E. Frydenberg, M. Ainley, and V. J. Russell, "Student motivation and engagement," *Schooling issues digest ; 2005/2*, 2005, 16p. [Online]. Available: http://www.dest.gov.au/sectors/school_education/publications_resources/schooling_issues_digest/documents/schooling_issues_digest_motivation_engagement_pdf.htm

[10] N. Bryan-Kinns and P. G. Healey, "Decay in collaborative music making," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, Paris, France, 2006, pp. 114–117. [Online]. Available: http://www.nime.org/proceedings/2006/nime2006_114.pdf

[11] G. Leslie, "Measuring musical engagement," *UC San Diego: Music*, 2013.

[12] M. E. Barrera, M. H. Rykov, and S. L. Doyle, "The effects of interactive music therapy on hospitalized children with cancer: a pilot study," *Psycho-Oncology*, vol. 11, no. 5, pp. 379–388, 2002.

[13] J. M. Standley and S. B. Hanser, "Music therapy research and applications in pediatric oncology treatment," *Journal of Pediatric Oncology Nursing*, vol. 12, no. 1, pp. 3–8, 1995.

[14] P. Keyani, G. Hsieh, B. Mutlu, M. Easterday, and J. Forlizzi, "Dancealong: supporting positive social exchange and exercise for the elderly through dance," in *CHI'05 extended abstracts on Human factors in computing systems*. ACM, 2005, pp. 1541–1544.

[15] J. Music and R. Murray-Smith, "Virtual hooping: teaching a phone about hula-hooping for fitness, fun and rehabilitation," in *Proceedings of the 12th international conference on Human computer interaction with mobile devices and services*. ACM, 2010, pp. 309–312.

[16] P. Riley, N. Alm, and A. Newell, "An interactive tool to promote musical creativity in people with dementia," *Computers in Human Behavior*, vol. 25, no. 3, pp. 599–608, 2009.

[17] K. Stensæth and E. Ruud, "An interactive technology for health: New possibilities for the field of music and health and for music therapy? a case study of two children with disabilities playing with "orfi"," *Series from the Centre for Music and Health*, vol. 8, no. 7, pp. 39–66, 2014.

[18] A. Gabrielsson and P. N. Juslin, *Emotional expression in music*. Oxford University Press, 2003.

[19] P. N. Juslin and P. Laukka, "Expression, perception, and induction of musical emotions: A review and a questionnaire study of everyday listening," *Journal of New Music Research*, vol. 33, no. 3, pp. 217–238, 2004.

[20] L.-L. Balkwill, W. F. Thompson, and R. Matsunaga, "Recognition of emotion in japanese, western, and hindustani music by japanese listeners," *Japanese Psychological Research*, vol. 46, no. 4, pp. 337–349, 2004.

[21] D. Huron, *Sweet anticipation: Music and the psychology of expectation*. MIT press, 2006.

[22] J. D. Lane, S. J. Kasian, J. E. Owens, and G. R. Marsh, "Binaural auditory beats affect vigilance performance and mood," *Physiology & behavior*, vol. 63, no. 2, pp. 249–252, 1998.

[23] A. Long, "Involve me: Using the orff approach within the elementary classroom," *Awards for Excellence in Student Research and Creative Activity*, no. 4, 2013.

[24] S. Hallam, "The power of music: Its impact on the intellectual, social and personal development of children and young people," *International Journal of Music Education*, vol. 28, no. 3, pp. 269–289, 2010.

[25] B. Osborn, "Understanding through-composition in post-rock, math-metal, and other post-millennial rock genres," *Music Theory Online*, vol. 17, no. 3, 2011.

[26] D. Watson, L. A. Clark, and A. Tellegen, "Development and validation of brief measures of positive and negative affect: the panas scales." *Journal of personality and social psychology*, vol. 54, no. 6, p. 1063, 1988.

[27] A. M. Sanal and S. Gorsev, "Psychological and physiological effects of singing in a choir," *Psychology of Music*, vol. 42, no. 3, pp. 420–429, 2014.

[28] J. Brooke, "Sus: a retrospective," *Journal of usability studies*, vol. 8, no. 2, pp. 29–40, 2013.

[29] B. Thorgaard, B. B. Henriksen, G. Pedersbæk, and I. Thomsen, "Specially selected music in the cardiac laboratory - an important tool for improvement of the wellbeing of patients," *European Journal of Cardiovascular Nursing*, vol. 3, no. 1, pp. 21–26, 2004.

[30] J. Sauro, "Measuring usability with the system usability scale (sus). 2011," *Available at: http://www. measuringusability. com/sus. php [22.04. 2013]*, 2015.

[31] T. S. Tullis and J. N. Stetson, "A comparison of questionnaires for assessing website usability," in *Usability Professional Association Conference*, 2004, pp. 1–12.

The musical influence on people's micromotion when standing still in groups

Alexander Refsum Jensenius, Agata Zelechowska, Victor Gonzalez Sanchez
University of Oslo, Department of Musicology, fourMs Lab
[a.r.jensenius, agata.zelechowska, v.e.g.sanchez]@imv.uio.no

ABSTRACT

The paper presents results from an experiment in which 91 subjects stood still on the floor for 6 minutes, with the first 3 minutes in silence, followed by 3 minutes with music. The head motion of the subjects was captured using an infra-red optical system. The results show that the average quantity of motion of standstill is 6.5 mm/s, and that the subjects moved more when listening to music (6.6 mm/s) than when standing still in silence (6.3 mm/s). This result confirms the belief that music induces motion, even when people try to stand still.

1. INTRODUCTION

It is commonly assumed that listening to musical sound, and particularly dance music with a clear pulse, "makes" us move. This assumption is to some extent supported by the literature in embodied music cognition [1,2], and there are also empirical studies of music-induced motion [3,4] or motion enhanced by music [5,6]. Many of these former studies have mainly focused on voluntary and fairly large-scale music-related body motion. As far as we know, there is little empirical evidence of music actually making people move when they try to remain at rest.

Our aim is to investigate the tiniest performable and perceivable human motion, what we refer to as *micromotion*. Such micromotion is primarily involuntary and performed at a scale that is barely observable to the human eye. Still we believe that such micromotion may be at the core of our cognition of music at large, being a natural manifestation of the *internal* motor engagement [7].

In our previous studies we have found that subjects exhibit a remarkably consistent level of micromotion when attempting to stand still in silence, even for extended periods of time (10 minutes) [8]. The measured standstill level of a person is also consistent with repeated measures over time [9]. These studies, however, were carried out on small groups of people (2–5), so we have been interested in testing whether these findings hold true also for larger groups.

In this paper we report on a study of *music-induced micromotion*, focusing on how music influences the motion of people trying to stand still. In order to answer that question, it is necessary to have baseline recordings of how much people move when standing still in silence. More specifically, this paper is aimed at answering the following questions:

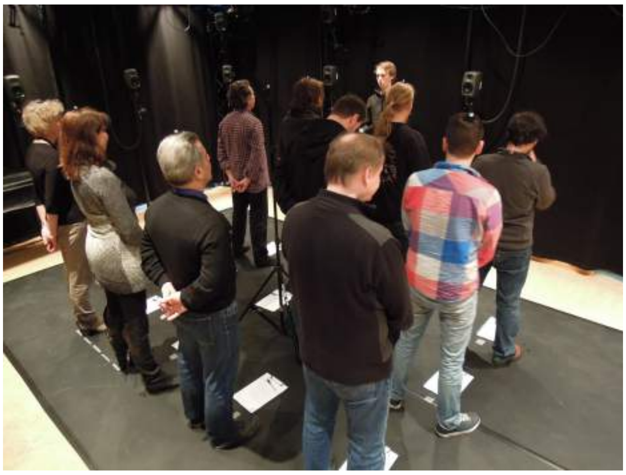


Figure 1. The setup for the "Norwegian Championship of Standstill." Each subject wore a reflective marker on the head, and one static marker was recorded from a standing pole in the middle of the space as a reference.

- How (much) do people move when trying to stand still?
- How (much) does music influence the micromotion observed during human standstill?

To answer these questions, we have started carrying out a series of group experiments under the umbrella name of the "Norwegian Championship of Standstill." The theoretical background of the study and a preliminary analysis have been presented in [10]. This paper presents a quantitative analysis of the data from the 2012 edition of our experiment series.

2. THE EXPERIMENT

The experiment was carried out in the fourMs motion capture lab at the University of Oslo in March 2012 (Figure 1).

2.1 Participants

A little more than 100 participants were recruited to the study, and they took part in groups consisting of 5-17 participants at a time (see Figure 1 for a picture of the setup). Not every participant completed the task and there were some missing marker data, resulting in a final dataset of

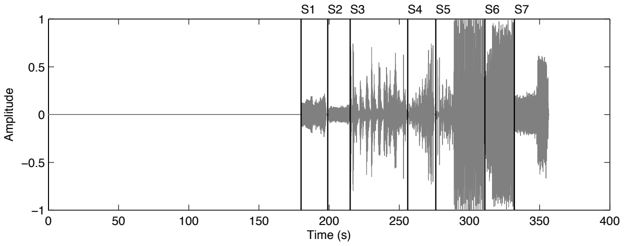


Figure 2. Waveform of the sound used throughout the experiment. Silence for the first 3 minutes, followed by 7 short music excerpts (S1–S7) ranging from non-rhythmic orchestral music to electronic dance music.

91 participants (48 male, 42 female, 1 unspecified). [1] The average age was 27 years (min = 16, max = 67). The participants reported quite diverse numbers for how many hours per week they spent listening to music (*M*=19, *SD*=15) and creating music (*M*=8, *SD*=8), reflecting that around half of the participants were music students.

2.2 Task

The task given to the participants was to attempt to stand as still as possible on the floor for 6 minutes in total, 3 minutes in silence and 3 minutes with music. They were aware that music would start after 3 minutes.

2.3 Sound stimulus

The sound file used as stimulus consisted of 3 minutes of silence, followed by 3 minutes of musical sound. There were 7 short musical excerpts, each with a duration of 20–40 seconds. The first musical excerpts were slow, non-rhythmic orchestral music, while the last ones were acoustical and electronic dance music. [2] As such, the rhythmic complexity and loudness increased throughout the experiment, as can be seen in Figure 2. The sound was played comfortably loud from a pair of Genelec 8020 loudspeakers and a Genelec 7050 subwoofer.

2.4 Motion capture

Each participant wore a reflective marker on his/her head, and its position was recorded using a Qualisys infrared motion capture system (Oqus 300) running at 100 Hz. We have previously shown that the spatial noise level of the system is considerably lower than that of human standstill [11].

Data was recorded and preprocessed in the Qualisys Track Manager, and the analysis was done in Matlab using the MoCap Toolbox [12].

To illustrate how the normalized position data looks like, Figure 3 shows plots of position on the three axes over time, as well as position spatial plots of the three planes.

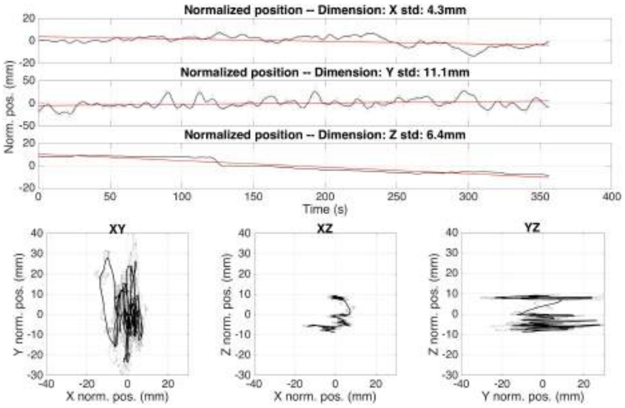


Figure 3. Example plots of the X (sideways), Y (front-back) and Z (updown) axes of the normalized position of a head marker. The light grey line is the raw data; the black line results from a ten-second smoothing; and the red line shows the linear regression (the trend) of the dataset.

3. RESULTS

3.1 Quantity of motion

To answer the question of *how much* people move, we calculated the *quantity of motion* (QoM) of each reflective marker by summing up all the differences of consecutive samples for the magnitude of the position vector, that is, the first derivative of the position:

$$QoM = \frac{1}{T}\sum_{n=2}^{N} \parallel p(n) - p(n-1) \parallel$$

where *p* is either the two-dimensional (XY axes—the horizontal plane) or three-dimensional (XYZ axes) position vector of a marker, *N* is the total number of samples and *T* is the total duration of the recording. The resultant QoM is measured in millimetres per second (mm/s).

In our previous studies [8, 9], we found QoM values in the range of 5–7 mm/s for a small group of people. Our new results confirm this range, with an average QoM of 6.5 mm/s (*SD* = 1.6 mm/s) over the complete recording, as summarised in Table 1. The lowest result was 3.9 mm/s (the winner!) and the highest was 13.7 mm/s. These values, however, included both the no-sound and sound conditions, so Table 1 also shows a breakdown of the values in these two conditions, as well as for the individual sound tracks. These differences will be further discussed in Section 3.5.

3.2 Motion over time

An interesting finding is that, for most participants, the quantity of motion did not change much over time, which can also be seen in the cumulative distance plots in Figure 4. There were a few extreme cases, but most participants had consistent linear motion distribution over time. Coefficient of determination (R-Squared) values were above 0.9 for most participants (mean R^2 = 0.94, s.d. R^2 = 0.0039 minimum R^2= 0.93).

[1] This paper is based on the complete dataset, while a subset was used for the qualitative analysis presented in [10].

[2] See `http://www.uio.no/english/research/groups/fourms/downloads/motion-capture/nm2012/` for detailed information about the music excerpts.

Table 1. Mean QoM values (in mm/s) for all sessions, in both no-sound and sound conditions, as well as for each of the individual music sections.

	No sound (3 min)	Sound (3 min)						
Part	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Mean QoM (mm/s)	6.5							
Mean QoM (mm/s)	6.3	6.6						
Mean QoM (mm/s)	6.3	6.2	6.5	6.7	6.5	6.6	6.9	6.7
Standard deviation	1.4	1.8	1.9	1.9	1.7	1.8	3.8	2.3

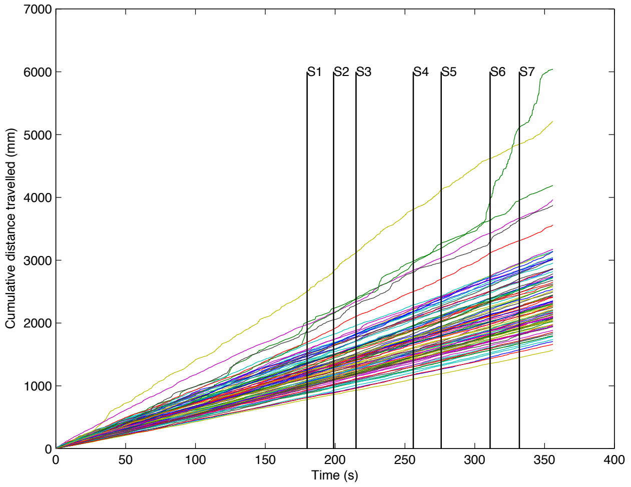


Figure 4. Cumulative distance travelled for all participants.

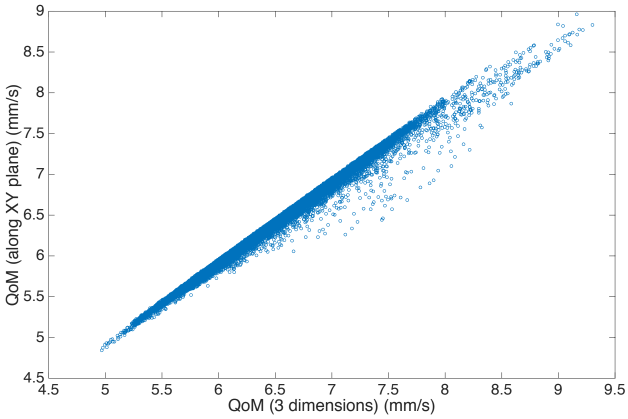


Figure 5. Scatter plot showing the linearity between QoM occurring in the horizontal (XY) plane and three-dimensional (XYZ) for the entire data set.

3.3 Horizontal Motion

To answer the question of *how* people move over time, we computed planar quantity of motion. The horizontal QoM (over the XY plane) was computed for all participants in order to further test the differences between conditions and stimuli. The mean horizontal QoM was found to be 6.4 mm/s for the entire 6-minute recording (*SD* = 1.5 mm/s). This value is only marginally smaller than the 6.5 mm/s found for the 3D QoM, suggesting that most motion, in fact, occurred in the XY plane. The relation between horizontal and 3D motion can also be seen in Figure 5.

3.4 Vertical Motion

To investigate the level of vertical motion, we also calculated QoM along the Z-axis. The mean vertical QoM across participants and conditions was 0.73 mm/s (*SD* = 0.52 mm/s), considerably smaller than the horizontal QoM reported above. This can also be seen in plots of the vertical motion (Figure 7) and in the frontal (YZ) plane (Figure 6), in which the bulk of motion in the Z axis is below 1 mm/s.

When looking at the differences between conditions, the mean vertical QoM during the no-sound segment of the trials was found to be 0.69 mm/s, while for the sound segment it was 0.77 mm/s.

3.5 Influence of sound on motion

For the 3-minute parts without sound we found an average QoM of 6.3 mm/s (*SD* = 1.4 mm/s), as opposed to 6.6 mm/s (*SD* = 2.2 mm/s) for the part with sound. This is not a dramatic difference, but shows that the musical stimuli did influence the level of standstill. A paired sample t-test was conducted to evaluate statistical significance of the observed differences between sound and no-sound conditions across the sample group. The results indicate the differences in means for three-dimensional QoM were significant for a 95% confidence interval (t = 2.48, p = .015).

Differences in the planar QoM between the sound (6.5 mm/s) and no-sound (6.2 mm/s) segments of the experiment were also statistically significant (t = 2.5, $p < .05$), although not considerably larger than those observed from 3D QoM.

These observed differences between sound and no-sound conditions were further explored by conducting a k-means cluster analysis of both 3D and 2D QoM for the entire data set. Using instantaneous QoM as a predictor, two clusters were identified by the implemented algorithm, although, as seen in the silhouette plot in Figure 8, most points in the clusters have silhouette values smaller than 0.3. This indicates that the clusters are not entirely separated, which could be due to the homogeneity of the sample group and the continuous nature of the musical stimuli.

The results are even clearer when looking at the individual stimuli in Table 1, with a QoM of 6.9 mm/s for the electronic dance music sample (#7) and 6.7 mm/s for the salsa excerpt (#8). As such, the results confirm our expectation that “music makes you move.” Even though the result may not be very surprising in itself, it is interesting to see that even in a competition during which the partici-

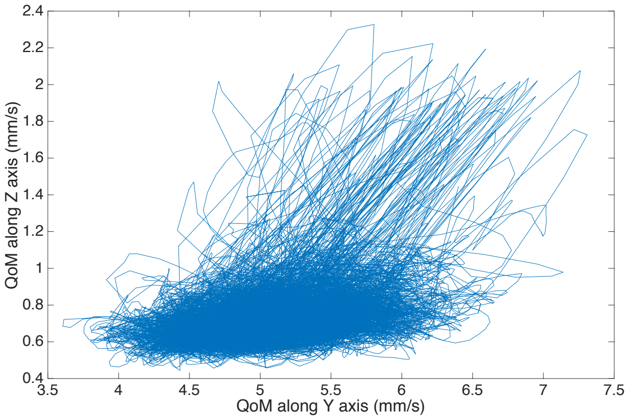


Figure 6. Plot showing QoM in the vertical plane (YZ) for the entire data set. The majority of the motion along this direction was below 1 mm/s.

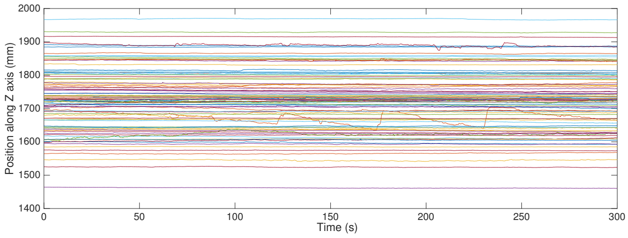


Figure 7. Instantaneous position of the marker along the Z axis (vertical direction).

pants actively try to stand absolutely still, the music has an influence on their motion in what can be termed ”micro” level.

3.6 Age, Height and Gender

We found a significant negative correlation between the average QoM results and the participants' age. Generally, younger participants tended to move more (r = -.278, $p <$ .01), both in the no-sound (r = -.283, $p <$ .01) and sound conditions (r = -.255, $p <$ .05). From the reported demographic information, we also found that the younger participants listened to music more frequently (r = -,267, $p <$ .05) and exercised more (r = -.208, $p <$ .05). The younger participants also reported feeling less tired during the experiment (r = -.35, $p <$ .001), subjectively experienced greater motion (r = -.215, $p <$ .05), and also reported moving more when sound was being played (r = -.22, $p <$ .05).

Unexpectedly, the QoM results did not correlate with the participants' height, which was estimated by calculating the average of each participant's Z-axis values. Due to a lower centre of mass, we would have expected to see shorter people with lower QoM results. However, the winner was 192 cm tall, while the runner-up was 165 cm.

Also, there were no significant differences in performance between male and female participants (no difference in average QoM, QoM in silence, QoM in music or QoM between both conditions).

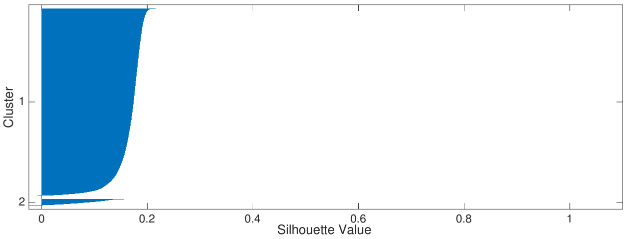


Figure 8. Silhouette plot from k means clustering analysis of QoM along the XY plane for the entire data set.

3.7 Effects of group, posture and physical activity

Aiming to evaluate the effects of standing strategies and postures, the participants were allowed to choose their standing posture during the experiment. In the post-experiment questionnaire they were asked to self-report on whether they were standing with their eyes open or closed, and whether they had their knees locked. The majority of the participants reported that they stood with open eyes (N = 62 versus N = 4 for closed eyes, and N = 8 for those who switched between eyes opened and closed during the experiment). Furthermore, 33 of the participants reported standing with locked knees, 31 switched between open and locked knees and 10 reported standing with their knees open. A 1-way ANOVA was performed to test if any of these factors influenced the average QoM of the participants, but showed no statistically significant results.

Interestingly, the participants who reported greater amount of time spent doing physical exercise tended to move more during the experiment (r = -.299, $p <$ .01). This tendency was particularly evident during the no-sound section (r = -.337, $p <$ .01), but it was also observed during the sound section (r = -.251, p = $<$ .05).

Additionally, we compared the average QoM results for all conditions (no-sound, sound, average no-sound and sound, and computed difference between sound and no-sound conditions) between groups of participants. Participants were split into 10 groups of varying age (F(8, 82) = 3.43, $p <$ .05), experience with performing , composing or producing music (F(8, 82) = 2.4, $p <$ .05), size (min = 5, max = 17) and the proportion of gender. We found no statistically significant differences between groups across these characteristics.

3.8 Subjective experience of motion

After taking part in the experiment the participants were asked to estimate how much they moved, to what extent the music influenced their movement, and how tiresome the experience felt. Overall, the self-reported tiredness showed some correlation with self-reported motion (r = -.44, $p <$ .001) and with the self-reported experience of moving more to music (r = -.289, $p <$ .01). The kinematic data confirmed this sensation: the more tired the participants felt, the more they moved to music (r = -.228, $p <$ .05) and the greater was the difference in motion to sound compared to the no-sound conditions (r = -.311, $p <$ .01). More importantly, although the subjective experience of motion did not correlate with the measured level of motion, the participants who reported moving more to music did move more during the sound condition when compared to the no-sound condition (r = -.239, $p < .05$ for the difference in QoM between music and silence).

4. CONCLUSIONS

This study was aimed at further exploring the magnitude of micromotion and the influence of music on human standstill, based on the preliminary work presented in [10]. Quantity of motion (QoM) was shown to be a sensitive measure of micromotion for the conditions under analysis. The computation of both three-dimensional and planar QoM showed that micromotion occurred mainly on the horizontal plane. Additionally, statistically significant differences were found between no-sound and sound conditions across the dataset. Two clusters were identified in the data through k-means cluster analysis, although most points in the clusters had silhouette values below 0.4. This could be due to the continuous nature of the sound stimuli and the small (although statistically significant) differences between conditions.

The analysis revealed some relationships between QoM data and the self-reported characteristics of physical activity and demographic information. People who exercised regularly found it more difficult to stand still. Moreover, younger participants tended to move more during both no-sound and sound conditions. These results may suggest that people who tend to be more active struggle to reach and maintain a complete standstill posture, although they might be able to stand normally for longer periods of time and with greater balance. The correlation found between self-reported tiredness and both self-reported and measured motion can not be considered conclusive and further studies will focus on a more in depth assessment of the effects of tiredness in combination with sound stimuli during standstill.

The fact that there were no significant QoM differences between the groups of participants, indicates that testing varying number of participants at once is a viable way to test our hypotheses. Future work will focus on studying larger sample groups and use different stimuli, with a focus on investigating in more depth how different musical features influence the micromotion of people standing still.

Acknowledgments

This research has been supported by the Norwegian Research Council (#250698) and Arts Council Norway.

5. REFERENCES

[1] M. Leman, *Embodied music cognition and mediation technology*. Cambridge, Mass.: MIT Press, 2008.

[2] R. I. Godoy and M. Leman, Eds., *Musical Gestures: Sound, Movement, and Meaning*. New York: Routledge, 2010.

[3] P. Toiviainen, G. Luck, and M. Thompson, "Embodied meter: Hierarchical eigenmodes in music-induced movement," *Music Perception*, vol. 28, no. 1, pp. 59–70, 2010. [Online]. Available: http://mp.ucpress.edu/content/28/1/59

[4] B. Burger, M. R. Thompson, G. Luck, S. Saarikallio, and P. Toiviainen, "Influences of Rhythm- and Timbre-Related Musical Features on Characteristics of Music-Induced Movement," *Frontiers in Psychology*, vol. 4, 2013. [Online]. Available: http://journal.frontiersin.org/article/10.3389/fpsyg.2013.00183/abstract

[5] M. Peckel, T. Pozzo, and E. Bigand, "The impact of the perception of rhythmic music on self-paced oscillatory movements," *Frontiers in Psychology*, vol. 5, Sep. 2014. [Online]. Available: http://journal.frontiersin.org/journal/10.3389/fpsyg.2014.01037/full

[6] F. Styns, L. van Noorden, D. Moelants, and M. Leman, "Walking on music," *Human Movement Science*, vol. 26, no. 5, pp. 769–785, 2007. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0167945707000589

[7] Y.-H. Su and E. Poppel, "Body movement enhances the extraction of temporal structures in auditory sequences," *Psychological Research*, vol. 76, no. 3, pp. 373–382, May 2012. [Online]. Available: http://link.springer.com/10.1007/s00426-011-0346-3

[8] A. R. Jensenius and K. A. V. Bjerkestrand, "Exploring micromovements with motion capture and sonification," in *Arts and Technology, Revised Selected Papers*, ser. LNICST, A. L. Brooks, Ed. Berlin: Springer, 2012, vol. 101, pp. 100–107. [Online]. Available: http://www.springerlink.com/content/j04650123p105646/

[9] A. R. Jensenius, K. A. V. Bjerkestrand, and V. Johnson, "How still is still? Exploring human standstill for artistic applications," *International Journal of Arts and Technology*, vol. 7, no. 2/3, p. 207, 2014. [Online]. Available: http://www.inderscience.com/link.php?id=60943

[10] A. R. Jensenius, "Exploring music-related micromotion," in *Body, Sound and Space in Music and Beyond: Multimodal Explorations*. Routledge, 2017, pp. 29–48, https://www.routledge.com/Body-Sound-and-Space-in-Music-and-Beyond-Multimodal-Explorations/Wollner/p/book/9781472485403.

[11] A. R. Jensenius, K. Nymoen, S. Skogstad, and A. Voldsund, "A Study of the Noise-Level in Two Infrared Marker-Based Motion Capture Systems," in *Proceedings of the Sound and Music Computing Conference*, Copenhagen, 2012, pp. 258–263. [Online]. Available: http://urn.nb.no/URN:NBN:no-31295

[12] B. Burger and P. Toiviainen, "MoCap Toolbox - A Matlab toolbox for computational analysis of movement data," in *Proceedings of the Sound and Music Computing Conference*, 2013, pp. 172–178. [Online]. Available: https://jyx.jyu.fi/dspace/handle/123456789/42837

SECCIMA: Singing and Ear Training for Children with Cochlear Implants via a Mobile Application

Zhiyan Duan, Chitralekha Gupta, Graham Percival, David Grunberg and Ye Wang
National University of Singapore
{zhiyan,chitralekha}@u.nus.edu, graham@percival-music.ca, {grunberg,wangye}@comp.nus.edu.sg

ABSTRACT

Cochlear implants (CI) can restore part of the hearing of people with severe hearing loss, but these devices are far from perfect. Although this technology allows many users to perceive speech in a quiet room, it is not so successful for music perception. Many public spaces are awash with music, but many CI users do not find music enjoyable or reassuring. This brings multiple challenges to their everyday lives, especially in a world that is saturated with music. Research shows that music training can improve music perception and appreciation for CI users. However, compared to multiple computer-assisted solutions for language training, few such systems for music training are available for children with CI. Existing systems are either targeting a different audience or have complex interfaces that are not suitable for children. In this study, we examined the design limitations of a prior application (MOGAT) and developed a new system with more suitable interfaces. The new system, SECCIMA, was crafted through an iterative design process that involved 16 participants, and the final system was evaluated and compared against MOGAT with another 12 participants. Our results show that SECCIMA is more intuitive and user-friendly than MOGAT.

1 Introduction

Cochlear implants (CI) are electronic devices implanted surgically that directly stimulate the auditory nerve and restore hearing for the deaf. Since their debut in the 1970s, over 120,000 people worldwide have received implants; however, CI are generally optimized for speech rather than music perception [1]. Many CI devices currently in use are limited to 22 electrodes, which each electrode corresponding to one frequency band of vocoder-processed audio. The resulting audio is very dissimilar to how audio is perceived without CIs, and familiar melodies often become completely unrecognizable [1] . Appreciating music can therefore be quite challenging for CI users, who usually have difficulty perceiving both melody and timbre, and do not rank “listening to music” as a pleasurable activity [2, 3]. In a world saturated with music, an inability to appreciate music can dramatically affect one’s quality of life [4]. Many adult CI users consider music “noisy” or “disappointing”, and listen to music less often than they did prior to hearing loss [4]. However, exposure to music is inevitable in public places. If music is foreign, then one’s quality of life in public places will plummet. Greater understanding, even if not complete appreciation of music, will therefore lessen anxiety and lead to a higher quality of life.

Music Training is an effective way to improve music perception and appreciation [4, 5]. However, it is manpower intensive and is conducted periodically, not on a daily basis. Each child can only receive a limited amount of individual attention with current training procedures. It would be beneficial to use mobile devices to help these children engage in learning anytime and anywhere.

In [6], a computer system for adults with CI explored various aspects of music such as timbre, rhythm, mixing and composition. Given its target audience (i.e. adults) and complexity, this system is less than ideal for children. Another work specifically targeted children with CI: MOGAT [7], but while the basic idea behind this work was inspiring, we were concerned that the interface might not be optimal for the target audience. Designing for users — especially children — with disabilities requires extra effort and consideration. Some initial consultations with teachers and therapists gave weight to our concerns, which led to the current project: designing games with user interfaces that are friendly to children with CI.

In this study, we designed a game-based ear training mobile application (SECCIMA) that facilitates the training of musical pitch perception and production skills of children with CI. The game was crafted through an iterative design process that involved 16 children with CI. After the game was finalized, it was compared to MOGAT with an evaluation consisting of 12 new children. We report on our iterative design process, evaluation and comparison results, as well as insights gained while observing the user interaction. Some guidelines on designing user interfaces for children with CI are discussed as well.

The contributions of this work are **i)** designing a game-based music training system that caters to the needs of children with cochlear implants; **ii)** evaluating and comparing the system with a prior application using both objective and subjective measures; and **iii)** summarizing some guidelines for designing user interfaces for children with cochlear implants.

[1] The readers are encouraged to test this themselves with audio such as https://www.youtube.com/watch?v=iwbwhfCWs2Q.

2 Related Work

2.1 Musical Training

Musical training can have a powerful effect on music perception and appreciation. An excellent review of music training for CI users is given in [4]; crucially, it supports the idea that training has the potential to improve music perception and appreciation in this population.

A recent examination of the effect of musical training on CI users found that it improved rhythm, timbre, and melodic contour recognition [8]. These results were echoed in [5], showing that melodic contour identification performance improved sharply after four weeks of music training, and no significant decline was found in the follow-up study eight weeks after the training intervention.

2.2 Computer-aided Training Solutions

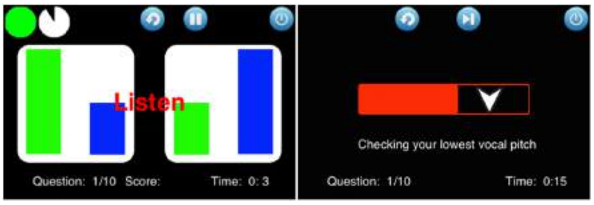
There are a number of computer-based **speech** training programs for CI users [9, 10]. However, two recent meta-reviews of computer auditory speech training cautioned that although many studies yielded positive results, the quality of the studies was low to moderate [11, 12]. Other projects explored new possibilities for interaction without performing any usability tests, such as [13], which used an interactive floor to combine movement-based playing and language acquisition , and [14], which allowed children to progress through a story by identifying phonemes.

There are few computer-based **music** training programs. The "Interactive Music Awareness Program" is a computer system for adults which allows users to explore various aspects of music [6], including melodies, rhythms, and texture. A 12-week study with 16 adult CI users showed that IMAP increased their ability to recognize musical instruments. Our main inspiration was "Mobile Game with Auditory Training for Children with Cochlear Implants" (MOGAT) [7]. This project provided three games: "Higher Lower", which prompted children to identify whether a pair of sounds were in ascending or descending pitch; "Vocal Matcher", which asked children to sing notes with specific pitches, and "Ladder Singer", which provided a pitch-graded karaoke interface. However, some flaws in the UI of this system caused problems when being used by children. These are discussed in Section 3.

2.3 Design for Children with Disabilities

In many cases, children with CIs are not as cognitively developed as children with typical hearing, with literacy, language, and behavioral issues due to the cumulative effect of the hearing loss before diagnosis, waiting time for the surgery, and recovery time after the implant [15, 16].

The specific design needs of deaf or hard of hearing children are discussed in two recent papers. [17] noted communication and behavioral difficulties during prototyping sessions, as well as the children being initially hesitant to explore the game being designed. [17] also confirmed that the children were very sensitive to visual changes, quickly noticing (and then fixating on) even minor changes to the background. [18] presented twenty-five guidelines for designing games for deaf children. Many of these guidelines are aimed at mitigating reduced language skills, addressing their sensitivity to visual changes, and mitigating their tendency to be distracted.



Checking your lowest vocal pitch

Figure 1: Two interfaces of MOGAT. Left: the "Higher-Lower" game. Right: the "Vocal Matcher" game.

3 Design Considerations

The goal of this study is to design a game-based training application on smartphones that can be used to improve pitch *perception* and *production* skills of children with CIs. Therefore, the application was developed to facilitate the following functions:

1. Play music notes audibly and display them visually;
2. Test players' perception of notes and give feedback;
3. Evaluate players' singing voice, and give feedback on the quality of the pitch produced by the player.

We began by examining MOGAT [7] (Figure 1) to identify design flaws. These are split into the following categories.

3.1 Simplicity

Children with CI are generally less developed in literacy and can be distracted by changing visual elements [17,18]. This means that UI designed for them must be straightforward and simple.

In MOGAT's "Vocal Matcher" game (Figure 1), the child had to first listen to a musical note and try to match that note by singing it. On the screen, there was an empty progress bar that gradually filled up as the child sustained the target pitch. When the pitch produced by the child was different from the target pitch, a small arrow appeared, showing the child how to adjust her pitch to approach the target. However, children sometimes adjusted their pitch in the wrong direction. Later examination revealed that there was an ambiguity in this interface. An arrow pointing down can be interpreted in two ways: either *your pitch is too high; please go lower*, or *compared to the target pitch, yours is too low; please go higher*.

Besides this ambiguity, the arrow design had another potential flaw. Normal progress bars seen in many other interfaces contain no arrows. First, the child was required to notice the arrow, and second, the child must learn the meaning of the arrow. This added to the cognitive load when children were already trying to sing the correct musical note and fill the progress bar.

Finally, screens in MOGAT included a status bar showing the number of completed questions, and a separate timer which counted up. These added little to the actual gameplay, and in the case of the timer could serve to distract the child by inducing them to watch the changing numbers

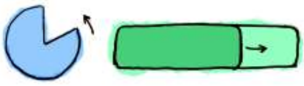


Figure 2: An inconsistency issue in MOGAT.

rather than performing the exercise.

3.2 Consistency/reinforcement

In MOGAT, there were a few inconsistencies between "Higher Lower" and "Vocal Matcher". While the former used a circle that gradually filled itself anti-clockwise during playback, the latter used a horizontal progress bar instead (Figure 2). Similarly, "Vocal Matcher" and "Ladder Singer" both represented musical notes with progress bars, but the latter used the vertical dimension to indicate the player's pitch while the former did not. These small inconsistencies may seem minor, but children with CI were easily confused when presented with such issues. Therefore, a consistent representation of key concepts is necessary.

Although consistency of the meaning of visual elements is important, that meaning does not need to be unique. It would be preferable if we can present information using multiple facets of visual elements without compromising the simplicity of the design.

3.3 Familiarity

Children with CI exhibit a degree of nervousness when facing unfamiliar environments [17]. Therefore, using concrete and familiar objects within the user interface would be a preferable approach when designing for this audience. The visual elements of MOGAT are mostly abstract shapes (e.g. circles, rectangles and bar graphs). While these visual elements look nice with their flat UI style, there were no metaphors between them and real-world objects with which the children are familiar. Therefore, children faced additional challenges in learning the relationship between the abstract graphical representations and the concrete concepts. For example, pitch difference was indicated with bar graphs, which was not taught until Primary 3 according to the local curriculum. Younger children were easily confused because these graphs were alien to them.

4 SECCIMA

In order to introduce familiarity into the new interface, we decided to use two real-world objects as the graphical elements, and attempted to find an analogy for each that was easy for the children with CI to comprehend. One was the *Ball*: when singing, the player's pitch controls the position of a ball. The player needs to drive the ball to a target position by singing the target pitch. The second was a *Balloon*: when singing at a target pitch, the player fills a balloon with air until it bursts; off-pitch singing makes the balloon lose air.

Our approach was to use the concept of reinforcement to exploit different *dimensions* of a single visual element, thus minimizing the cognitive load. Since the core underlying concept of SECCIMA is the pitch of music notes, we need to establish connections between the visual representation of the UI element and the pitch. The visual dimensions we utilized were: *color*, *position* and *movement*.

- *Color*: the red and blue end of the spectrum were used for higher and lower pitches respectively.
- *Position*: higher and lower pitches were positioned near the top and bottom of the screen respectively.
- *Movement*: upward/downward movements were used to represent increasing/decreasing pitch.

Utilizing multiple dimensions of a single object benefits the system in two ways. First, it establishes multiple connections between the target concept and the real-world analogy that children are already familiar with. Multiple connections increase the chances of them understanding the target concept. Moreover, since visual dimensions (color, position and movement) are simply attributes of familiar objects , the connections between these attributes and the object are easy to remember.

The game interfaces in SECCIMA are simple with fewer visual elements compared to MOGAT. We removed multiple indicators like progress bars, arrows and status bars. We used the position of glossy balls to indicate pitch. Children can therefore focus on just moving the ball to the correct position without having to worry about learning multiple indicators. In addition, the appearance and the function of the visual elements used are consistent across all the games. For example, the meaning of color, position, and movement of the glossy balls is the same for all the games.

5 Initial Game Design

There were three games in SECCIMA, namely *Xylophone*, *High/Low* and *Sing'NRoll* (Figure 3). The counterparts of High/Low and Sing'NRoll in MOGAT are Higher Lower and Vocal Matcher. We did not include a karaoke interface in SECCIMA, because the focus of SECCIMA was single pitch perception and production. We built the user interface around the *color*, *position* and *movement* of 8 glossy-looking balls. Each ball is correlated to a musical note with its color and position representing the pitch.

5.1 Xylophone

As the first game in SECCIMA, Xylophone serves to introduce the whole application. In this game, the player simply needs to tap on each glossy ball and listen to the corresponding music note. The purpose of this simple game is to lay the foundation for the rest of the games by helping the players learn the underlying analogies of the UI.

The Xylophone game begins by showing a random subset of the glossy balls. After listening to all notes by tapping all visible balls, the player is taken to the next level. As the levels progress, more balls are shown, ending with all 8 in their respective note positions. This enables to the child to gain familiarity with the interface.

5.2 High/Low

High/Low is the successor of the "Higher Lower" game of MOGAT, and provides training on pitch perception. The interface of High/Low has the same layout as Xylophone, with most of the octave lineup invisible. Two or three

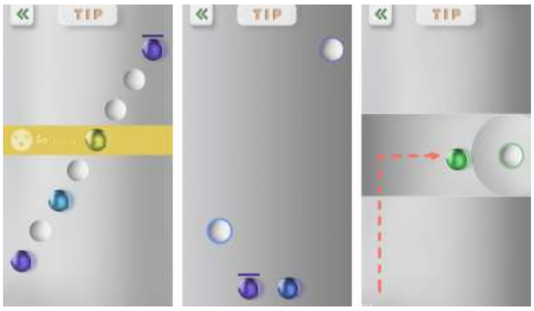


Figure 3: The initial design of the 3 games in SECCIMA: Xylophone, High/Low, and Sing'NRoll *(left to right).*

glossy balls are placed at the bottom of the screen, while the holes to which those balls belong are shown. The ball's color is the same as in the Xylophone game, while holes are surrounded by a halo of the corresponding color. Upon tapping a ball or a hole, the corresponding musical note is played. To win the game, the player needs to drag the balls to their corresponding holes, ideally by listening to the pitch of each ball and matching it with the correct hole.

As the levels of the game progress, the visual clues gradually disappear so as to challenge the player to rely more on her hearing than on her visual ability. From level three on, the halos around the holes are not colored anymore, and from level six onward, the colors of some balls disappear as well. The final level of this game has all the colors removed, forcing the player to solely rely on her hearing.

5.3 Sing'NRoll

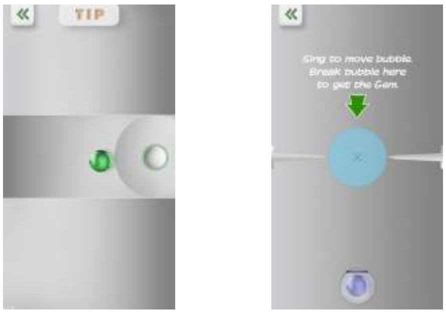
In Sing'NRoll, the player first listens to a musical note (i.e. the target) played by the application, then attempts to reproduce the pitch by singing. The smartphone captures the singing of the player and performs real-time audio analysis to extract the pitch using the YIN algorithm [19]. The user's pitch controls the vertical position of the ball. If the pitch matches the target pitch within a specific tolerance, the ball will be aligned with the hole on screen and start rolling towards the hole, hence the name Sing'NRoll. By sustaining the target pitch for a certain duration, the player can move the ball to the hole and win the game. If the player fails to hold the target pitch, the ball will gradually roll away from the hole, to discourage random guesses.

6 Iterative Design Process

After the prototype was completed, we followed an iterative design process (IDP) to improve the interface. Four rounds of iterative design were conducted, with 3 children in the first, 5 in the second, and 4 in the last two rounds.

6.1 Participants and Procedure

Participants of the user study were recruited from a local primary school dedicated to children with hearing impairment. There were 16 participants in total, with 13 boys and 3 girls. Their age range was 9 to 12 years. All of the participants were pre-lingually deaf, and 14 of them were congenitally deaf. Most of them (14 out of 16) had regular access to smartphones and used smartphones in general to play games. The age range of the participants' cochlear implant was between 10 to 135 months (mean 80.4, std 29.3). All participants had unilateral cochlear implants.



Sing to move bubble. Break bubble here to get the Gem.

(a) Before

(b) After

Figure 4: Sing'NRoll changes after round 1

The user studies were conducted in an empty, quiet classroom on the campus of the primary school where the children were recruited. Each session started with a 5-minute briefing, and then the participant played the game for a maximum duration of 15 minutes with no instructions or hints from the test conductors. Finally, a 5-minute feedback conversation was administered and questionnaire filled.

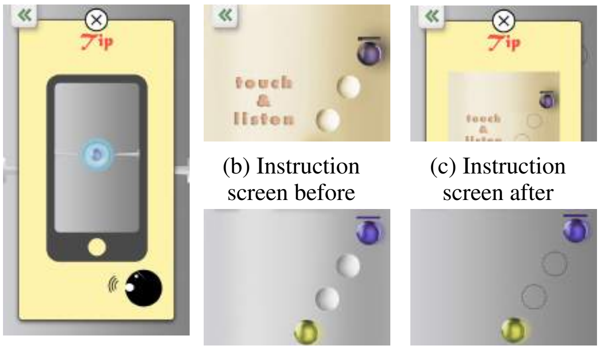
6.2 Round 1

All participants in this round had difficulty figuring out the Sing'NRoll game; they tried dragging the ball rather than singing. This might have been caused by the inconsistency between High/Low and Sing'NRoll, where the action to move a ball in the former was dragging but in the later switched to singing. Another issue was the lack of effective prompt to switch from one action to another. The instruction screen was skipped very quickly by most participants.

Based on these observations, we made the following changes to the design for the next round (Figure 4). For Sing'NRoll, we placed the ball inside a bubble to differentiate it from the balls in the High/Low game. When the player sang, the bubble moved vertically. If the player sustained singing at the target pitch, the bubble gradually grew until reaching some threshold and bursting. By changing the interface dramatically, our aim was to indicate that this game expected different inputs from High/Low. We also removed the hole entirely and replaced it with a target area to avoid any confusion between Sing'NRoll and High/Low. Another consequence of this change is that it avoided a potential confusion regarding movement. The original design used the Y-axis to indicate pitch (which we retain), and the X-axis to indicate completion. By abandoning X-axis movement and using the bubble's size as a completion metric, we clarify the meaning of movement in the game. Finally, a prompt message was added to remind the player to sing if she remained silent for more than 6 seconds.

6.3 Round 2

Neither the bubble nor the prompt message seemed to have any effect. Only two out of the five participants managed to figure out how to play the game. Although the prompt said explicitly "*Sing to move bubble. Break the bubble here to get the Gem.*", children did not seem to follow it. Young children with CI have difficulty understanding long



(a) Animated tip of Sing'NRoll

(b) Instruction screen before

(c) Instruction screen after

(d) Xylophone before

(e) Xylophone after

Figure 5: UI changes for round 2

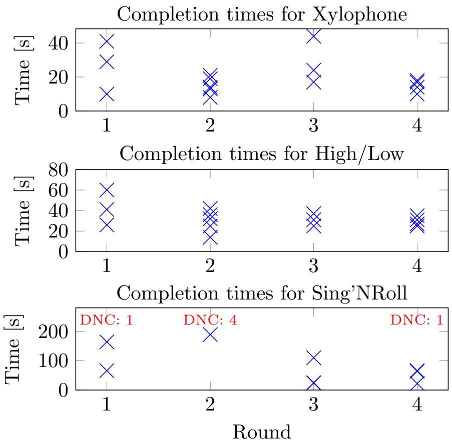


Figure 6: Completion times for games. "DNC" indicates that the child "did not complete" the exercise.

No.	Statement
Q1	I understood the instructions
Q2	The game was fun
Q3	The game was easy to play
Q4	I want to play this game again
Q5	I want to play this game at home
G1	Rate the Xylophone game
G2	Rate the High/Low game
G3	Rate the Sing'NRoll game

Table 1: Post-session questionnaire for participants.

sentences [17, 18], and we failed to adequately heed those warnings. The arrow pointing at the target area encouraged participants to tap on that area, which caused more confusion. A few participants also tried to play the game directly on the instruction screen, because it had a very similar look to the actual game and caused the confusion.

Another subtle consistency issue was observed. Since holes in Xylophone were not tappable, participants considered holes in High/Low to behave the same way. However, the later were designed to play the note upon tapping.

We made three changes after this round: 1) instruction screen of Sing'NRoll was changed from a static image to an animated demo (Figure 5a), 2) all instruction screens received a style update different from the actual games (Figure 5b, 5c), 3) holes in Xylophone and High/Low were given different appearances (Figure 5d, 5e).

6.4 Round 3 & Round 4

The animated cartoon instruction worked well. All four participants learned to play the Sing'NRoll game very quickly (three of them managed to finish the first level in under 30 seconds). The other two problems observed in round 2 were also resolved by the corresponding adjustments.

With no obvious flaws found in round 3, we added a scoreboard to Sing'NRoll as an extra motivational component to see if it can boost excitement.

No new problems were discovered in round 4, but there was also no effect of the scoreboard — all of the children quickly skipped over it. The scoreboard might contain more information than they would like to consume.

6.5 Objective Evaluation

During the iterative design process, participants' interactions with SECCIMA were filmed, from which we calculated the time taken for a participant to complete each game and used this time as an objective measure to quantify the intuitiveness and ease of use of the UI.

For Xylophone and Sing'NRoll, the completion time was defined as the time from a player entering the game to winning the first level of the game. The completion time of the High/Low game was defined differently because the first level of High/Low has color strokes around the targets and the participant can win the game by simple color matching. Therefore we defined the completion time of the High/Low game as the time lapsed from the player entering the game to the player winning the third level, where there are no color indications around the targets.

Figure 6 shows the completion time of all participants organized by round number and game. One participant in round 3 was excluded because the corresponding video recording was corrupted. Both the Xylophone game and the High/Low game were completed within a minute (many within 30 seconds). The completion time for the Sing'NRoll game was more interesting as this was the challenging part for the participants. In the first two rounds when there were no animated instructions, 5 out of 8 participants failed to complete the game. Those who managed to complete the game had struggled for quite some time (89, 164, 189 seconds) to succeed. After the introduction of the animated instruction in round three, the majority of the participants (7 out of 8) completed the game without much struggle. The completion time was reduced significantly as well, from more than two minutes to well under one minute.

6.6 Subjective Evaluation

Subjective measurements were collected via the post-session questionnaire (Table 1, Figure 7). The participants were asked to choose from a Likert scale of 5, with larger numbers corresponding to more positive answers. The responses to Q1 (*I understood the instructions*) and Q3 (*The game was easy to play*) exhibit a high correlation to the introduction of animated instruction, as they increase noticeably after round 2. This confirms our observation from the objective evaluation that the animated instruction did help the participants learn the game faster and make the

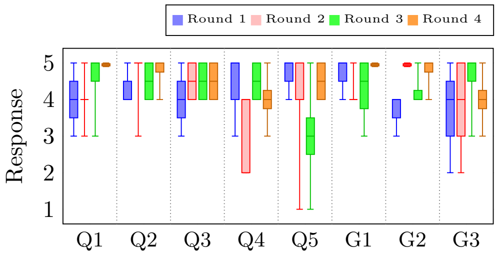


Figure 7: Questionnaire results for each round (box plots).

game easier. The scores of Q2 (*The game was fun*) increased marginally over time; however, the large variations in scores of Q4 and Q5 suggest that these improvements in enjoyment were not significant.

The participants were also asked to rate the three games on a Likert scale of 1 to 5. It can be seen that the scores of Xylophone (G1) and High/Low (G2) are noticeably higher than that of Sing'NRoll (G3). One possible reason is that in the Sing'NRoll game participants were facing more challenges, which may have attenuated their enjoyment. But even though Sing'NRoll is less preferred, its scores are on a upward trend, which points to the effectiveness of the adjustments to the system.

7 Comparison with MOGAT

SECCIMA was compared with MOGAT to assess its intuitiveness. In order to distinguish the two, we will prepend the game names with S- or M-, respectively. Both applications have two main components: a *listening game* (LG: M-Higher Lower vs S-High/Low) and a *singing game* (SG: M-Vocal Matcher vs S-Sing'NRoll in SECCIMA).

7.1 Participants, Setup and Procedure

A total of 12 participants were involved, out of which 7 were congenitally deaf, 4 pre-lingually deaf and 1 post-lingually deaf. The age range was 7 to 10 years. All participants were regular smartphone users and had used them for playing games. Time since cochlear implantation ranged from 20 to 84 months (mean 45.3, std 22.9).

For SECCIMA, the same equipment as in the iterative design process was used. For MOGAT, we used an iPod Touch (4$^{\text{th}}$ gen.). The same audio cable was used to connect the device with participants' implants.

All participants had **no prior exposure** to either interface. They were randomly divided into two groups, each using one of the apps. The same procedure as in the iterative design process was used, i.e. a pre-session briefing, 15 minutes of game play (video recorded), and a short discussion as well as a post-session questionnaire.

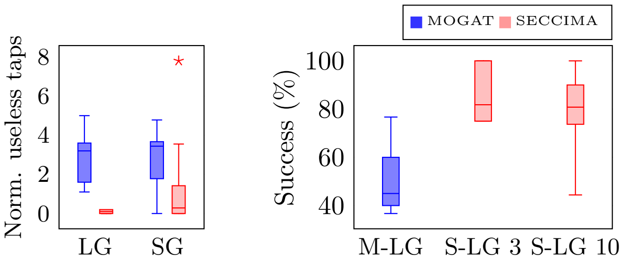
7.2 Objective Evaluation

Objective measures were derived from the video recordings. Since one participant opted out of video recording, we were left with 6 subjects in the SECCIMA group and 5 in the MOGAT group.

We counted the *unnecessary taps* (i.e. tapping on UI elements that are not tappable) by the participants in both groups (shown in Table 2), which was then divided by the total number of UI elements in respective games to normalize the difference in relative densities of screen layouts. We used this normalized number of unnecessary taps as an objective measure of the intuitiveness of the UI, with more unnecessary taps indicating a poor interface. As shown in Figure 8a, the normalized number of unnecessary taps in SECCIMA were significantly fewer than MOGAT in the listening game. Results for the singing game also favored SECCIMA, but the difference was not significant.

App	No. of UI elements	No. of useless taps
M-LG	10	11, 16, 32, 36, 50
S-LG	6	0, 0, 0, 0, 0, 0
M-SG	9	0, 16, 31, 33, 43
S-SG	7	0, 0, 0, 4, 10, 55

Table 2: Number of unnecessary taps from users



(a) Normalized unnecessary taps from users (*lower is better*). $p = \mathbf{0.001}, 0.51$.

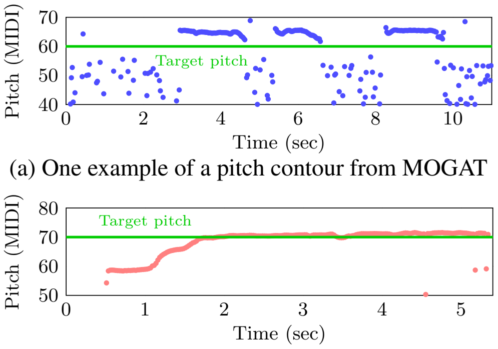
(b) Success rate in listening games (*higher is better*), S-H/L x: first x levels of S-High/Low. $p = \mathbf{0.006}, 0.06$.

Figure 8: Objective comparison (box plots).

Another objective measure we used was the percentage of successful attempts in the listening game. In both interfaces, the child was supposed to distinguish musical notes and to indicate which one is higher or lower. Choosing randomly would give a 50% successful rate. Figure 8b shows the successful rates for M-Higher Lower and S-High/Low. With a mean success rate of 50.67% (vs. 81.8 % and 80.8 % in SECCIMA), children playing M-Higher Lower were likely to be making selections randomly, indicating the user interface was not intuitive enough for the children to understand. Later levels of S-High/Low were more challenging because they involved more than two musical notes (only the first three levels involved two notes), but as student performance remained high, the interface of SECCIMA was demonstrated to be more effective.

In the singing game, **none** of the 6 children using MOGAT could figure out how to play the game on their own until the teacher intervened. On the contrary, 3 out of 6 children in the SECCIMA group managed to figure it out on their own, and another one kept blowing air instead of singing. Only two of them failed to understand that they were supposed to make sound. Although not as good as in the iterative design study (in which all but one child in rounds 3 and 4 figured it out), this result was not surprising considering the younger age of this group (7–10 vs. 9–12).

In terms of feedback, the children's pitches in the MOGAT group were mostly monotonic, suggesting that they did not fully understand the feedback. In the SECCIMA group, more children were varying their pitches in response



(a) One example of a pitch contour from MOGAT

(b) One example of a pitch contour from SECCIMA

Figure 9: Comparison of two example pitch contours.

to the feedback. As an example, Figure 9a and 9b show two typical pitch contours from users of MOGAT and SECCIMA respectively. The pitch contour from MOGAT was mostly monotonic and did not move towards the target pitch over time. On the other hand, the one from SECCIMA slid nicely towards the target pitch and stayed there.

For the comparison between the two *singing* games, "completion time" was not used as a comparing metric because the game designs were too different to compare fairly. For example,M-Vocal Matcher did not penalize the user for incorrect pitch, which made it much easier than S-Sing'NRoll, because the latter imposed a penalty for incorrect pitch and therefore was harder to win. For the same reason, success rate was not used either.

7.3 Subjective Evaluation

Subjective measurements were collected via the post-session questionnaire (questions in Table 1, results in Figure 10).

The children indicated that SECCIMA was easier to understand and play than MOGAT (Q1 and Q3). It is worth noting that some of the children playing MOGAT indicated they understood the game fully (by rating Q1 with a score of 5), although they did **not** play the game as they were supposed to. In fact, none of the children in the MOGAT group was able to figure out how to play the singing game (Vocal Matcher) after a long struggle, and the teacher had to intervene. Discussions with their teacher confirmed that they tend to say *Yes* to those questions they do not understand. This could potentially bring inaccuracies to the results. Researchers working on comparisons between people with different ages or mental capabilities may need to take this potential bias into consideration.

The children's teacher who helped us conduct the study provided some feedback on both interfaces. She mentioned that MOGAT was less intuitive. For example, the word *Select* used in MOGAT was not in the children's vocabulary, and the children had no exposure to bar graphs until their later stage in primary school. In contrast, SECCIMA's colored balls were simpler and easier to understand. She specifically pointed out that the arrows (see Figure 1) used by Vocal Matcher in MOGAT were confusing, even to her as an adult. Children were tapping on the flashing arrow even though it was not tappable. On the other hand, the animation used in SECCIMA was more intuitive.

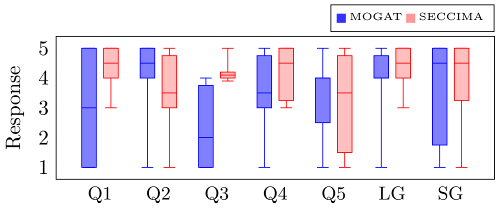


Figure 10: Questionnaire results (*higher is better*), see text for details about possible inaccuracies (box plots). $p = 0.21, 0.58, \mathbf{0.03}, 0.39, 0.74, 0.49, 0.76$.

8 Discussion

The interfaces for SECCIMA games were simpler than MOGAT, and the visual elements were consistent throughout all the games. The lower number of unnecessary taps and the higher success rate in SECCIMA confirms our hypothesis that increasing the simplicity and the consistency of the games made them easy to follow. Additionally, the reinforcement considerations resulted in a step-by-step learning process in SECCIMA. The lesson about color and position of the balls in the first game helped the students understand how to drag the balls to correct positions in the second game. Similarly, lessons from each level were being applied to play the next levels. The reduction in unnecessary taps and more-than-chance success rate of SECCIMA shows that the children were learning the games, and not randomly playing them. Finally, in SingN'Roll, familiarity with floating bubbles may have helped the children to understand the feedback on how to change their pitch. Thus, it was a compound effect of all design considerations that made the interface more intuitive and less confusing.

We offer a few recommendations for researchers working with children with CI. **(1) Test early, test often**: No matter how many hours are spent discussing designs in the lab, something unexpected will always happen when testing with children. That said, it is important to **(2) carefully balance the number of children in the design vs. evaluation phases**. There are usually limited number of CI users in each city, and parents and caregivers are appropriately cautious about children's participation in research studies.

It might be helpful to **(3) involve participants outside of the target population**. While the real evaluation should still be done with the target group, some of the flaws could have been detected by children with normal hearing.

It is important to **(4) reduce the amount of text and use more visual displays**, provided those visuals displays do not cause distractions. **(5) When text is necessary, words and phrases are more preferable then sentences**, because these children have difficulties comprehending longer text. It is worth noting that even the use of imagery and animation should be controlled to minimize cognitive load. These findings are consistent with those of [17, 18].

We observed that many young children with CI are not able to express themselves effectively (also noted in [16, 17]); forming sentences can be difficult for them. Therefore, **(6) obtaining verbal feedback from them takes time and patience** (as with adults with cognitive disabilities [20]). As noted earlier in the comparison study, some children indicated that they completely understood the instructions while they actually did not. When collecting subjective feedback from these children, this "*nodding phenomenon*" should be taken into consideration to avoid potential bias. We therefore suggest to **(7) use objective measurements in addition to subjective feedback**, such as success rate, number of unnecessary taps, and completion times.

9 Conclusion

We designed SECCIMA to facilitate music training of children with cochlear implants using an iterative design process. The observations and design decisions in each iteration were documented. Both objective and subjective evaluations were performed to assess the effectiveness of the design decisions and the ease of use of the interface. The evaluation result shows that the adjustments between iterations improved the usability and made the interface easier to use. A comparison with an existing work was performed and showed that our system is more intuitive and easier for the children to use. The insights and guidelines summarized from this study may be helpful to those who are interested in designing applications for children with CI.

Acknowledgments

The authors would like to thank Ms. Joan Tan for facilitating the user study as well as many excellent suggestions to our project. We are grateful to Ms. Terry Theseira and her colleagues and students from Canossian School for assistance and participation in the user studies. This project was partially funded by a research grant R-252-000-597-281 from National Research Foundation in Singapore.

10 References

[1] B. S. Wilson and M. F. Dorman, "Cochlear implants: a remarkable past and a brilliant future," *Hearing research*, vol. 242, no. 1, pp. 3–21, 2008.

[2] H. J. McDermott, "Music perception with cochlear implants: a review," *Trends in amplification*, vol. 8, no. 2, pp. 49–82, 2004.

[3] C. J. Limb and A. T. Roy, "Technological, biological, and acoustical constraints to music perception in cochlear implant users," *Hearing Research*, vol. 308, pp. 13 – 26, 2014.

[4] V. Looi, K. Gfeller, and V. Driscoll, "Music appreciation and training for cochlear implant recipients: a review," in *Seminars in hearing*, vol. 33, no. 4, 2012, p. 307.

[5] Q.-J. Fu, J. J. Galvin, X. Wang, and J.-L. Wu, "Benefits of music training in mandarin-speaking pediatric cochlear implant users," *Journal of Speech, Language, and Hearing Research*, pp. 1–7, 2014.

[6] R. M. van Besouw, B. R. Oliver, M. L. Grasmeder, S. M. Hodkinson, and H. Solheim, "Evaluation of an interactive music awareness program for cochlear implant recipients," *Music Perception: An Interdisciplinary Journal*, vol. 33, no. 4, pp. 493–508, 2016.

[7] Y. Zhou, K. C. Sim, P. Tan, and Y. Wang, "Mogat: mobile games with auditory training for children with cochlear implants," in *ACM Multimedia*. ACM, 2012, pp. 429–438.

[8] B. Petersen, M. V. Mortensen, M. Hansen, and P. Vuust, "Singing in the key of life: A study on effects of musical ear training after cochlear implantation." *Psychomusicology: Music, Mind, and Brain*, vol. 22, no. 2, p. 134, 2012.

[9] Q.-J. Fu and J. J. Galvin III, "Maximizing cochlear implant patients' performance with advanced speech training procedures," *Hearing research*, vol. 242, no. 1, pp. 198–208, 2008.

[10] ——, "Computer-assisted speech training for cochlear implant patients: Feasibility, outcomes, and future directions," in *Seminars in hearing*, vol. 28, no. 2. NIH Public Access, 2007.

[11] H. Henshaw and M. A. Ferguson, "Efficacy of individual computer-based auditory training for people with hearing loss: A systematic review of the evidence," *PLoS ONE*, vol. 8, no. 5, p. e62836, 05 2013.

[12] R. Pizarek, V. Shafiro, and P. McCarthy, "Effect of computerized auditory training on speech perception of adults with hearing impairment," *SIG 7 Perspectives on Aural Rehabilitation and Its Instrumentation*, vol. 20, no. 3, pp. 91–106, 2013.

[13] O. S. Iversen, K. J. Kortbek, K. R. Nielsen, and L. Aagaard, "Stepstone: An interactive floor application for hearing impaired children with a cochlear implant," in *Interaction Design and Children*, ser. IDC '07. New York, NY, USA: ACM, 2007, pp. 117–124. [Online]. Available: http://doi.acm.org/10.1145/1297277.1297301

[14] S. Cano, V. Peñeñory, C. A. Collazos, H. M. Fardoun, and D. M. Alghazzawi, "Training with Phonak: Serious Game As Support in Auditory – Verbal Therapy for Children with Cochlear Implants," in *ICTs for Improving Patients Rehabilitation Research Techniques*, ser. REHAB '15. New York, NY, USA: ACM, 2015, pp. 22–25. [Online]. Available: http://doi.acm.org/10.1145/2838944.2838950

[15] R. Calderon and M. Greenberg, "Social and emotional development of deaf children: Family, school, and program effects," *The Oxford handbook of deaf studies, language, and education*, vol. 1, pp. 188–199, 2011.

[16] S. C. Theunissen, C. Rieffe, M. Kouwenberg, L. J. De Raeve, W. Soede, J. J. Briaire, and J. H. Frijns, "Behavioral problems in school-aged hearing-impaired children: the influence of sociodemographic, linguistic, and medical factors," *European child & adolescent psychiatry*, vol. 23, no. 4, pp. 187–196, 2014.

[17] L. E. Potter, J. Korte, and S. Nielsen, "Design with the deaf: Do deaf children need their own approach when designing technology?" in *Interaction Design and Children*, ser. IDC '14. New York, NY, USA: ACM, 2014, pp. 249–252.

[18] A. Melonio and R. Gennari, "How to design games for deaf children: Evidence-based guidelines," in *2nd International Workshop on Evidence-based Technology Enhanced Learning*, ser. Advances in Intelligent Systems and Computing. Springer International Publishing, 2013, vol. 218, pp. 83–92.

[19] A. De Cheveigné and H. Kawahara, "Yin, a fundamental frequency estimator for speech and music," *J. of the Acoust. Soc. of America*, vol. 111, no. 4, pp. 1917–1930, 2002.

[20] S. Johansson, J. Gulliksen, and A. Lantz, "User participation when users have mental and cognitive disabilities," in *Proceedings of the 17th International ACM SIGACCESS Conference on Computers & Accessibility*, ser. ASSETS '15. New York, NY, USA: ACM, 2015, pp. 69–76. [Online]. Available: http://doi.acm.org/10.1145/2700648.2809849

STACKED CONVOLUTIONAL AND RECURRENT NEURAL NETWORKS FOR MUSIC EMOTION RECOGNITION

Miroslav Malik*‡, Sharath Adavanne*§, Konstantinos Drossos§, Tuomas Virtanen§, Dasa Ticha‡, and Roman Jarina‡

‡Department of Multimedia and Information-Communication Technologies, University of Zilina, Slovakia
firstname.lastname@fel.uniza.sk
§Audio Research Group, Department of Signal Processing, Tampere University of Technology, Finland
firstname.lastname@tut.fi

ABSTRACT

This paper studies the emotion recognition from musical tracks in the 2-dimensional valence-arousal (V-A) emotional space. We propose a method based on convolutional (CNN) and recurrent neural networks (RNN), having significantly fewer parameters compared with state-of-the-art method for the same task. We utilize one CNN layer followed by two branches of RNNs trained separately for arousal and valence. The method was evaluated using the "MediaEval2015 emotion in music" dataset. We achieved an RMSE of 0.202 for arousal and 0.268 for valence, which is the best result reported on this dataset.

1. INTRODUCTION

Music has been used to transfer and convey emotions since the initial era of human communication [1, 2]. In the field of music signal processing, either perceived or induced emotion (see [3] for details) has been proven to be a strong and content-oriented attribute of music and has been employed for categorization tasks. For example, published works focus on emotion recognition from music in order to provide an alternative and content-oriented categorization scheme to the typical "Artist/Band/Year" one [4, 5].

There are two main categories of models for emotions; i) the discrete, and ii) the dimensional models. Discrete models of emotion employ a discrete group of words to model emotions. The basic emotions model [6] and the Hevner's list [7] are the important representative examples of this group. The main drawback of discrete models of emotion is the subjective perception of the meaning of each word that is employed to describe an emotion. This leads to confusion when different words are utilized for the same emotion, e.g. "Joy"–"Enjoyment" and "Cheerfulness"–"Happiness" [2, 8]. Despite this confusion, discrete models are heavily used in works where the focus is on a specific emotion or emotional condition [9].

This categorical approach is also employed in the audio mood classification (AMC) task, under the music information retrieval evaluation exchange (MIREX) [10]. At AMC, the participants were requested to propose methods and systems capable of classifying audio files into five discrete emotion classes, using one class for the whole audio clip. The five classes contained five to seven adjectives and these adjectives were derived from online tags and were not based on some of the accepted categorical models of emotion. On average, the accuracy of the proposed methods and systems was below 70% [11, 12].

Dimensional models represent emotion in a continuous N-dimensional space. Usually, $N = 2$ and one dimension corresponds to arousal and the other to valence [13]. The emotion in a song is usually not spread equally through the time and it varies around some bias value. The bias value usually stays unchanged for a significant fraction of the duration of the song [14]. This property can be hardly observable in the categorical approach, but with the dimensional approach, the continuous changes can be captured more easily.

A dimensional model was employed for the emotion in music (EiM) challenge, organized under the MediaEval benchmark [15]. To the best of authors' knowledge, MediaEval along with MIREX are the two widely known international scientific challenges for music emotion recognition. The MediaEval challenge consisted of two tasks, one focusing on static and the other focused on the dynamic presence of emotion in music. In the second (dynamic) task of the EiM, the methods proposed were based on support vector regression (SVR) [16], continuous conditional neural fields [17], continuous conditional random fields [18], feed-forward neural networks [19], and recurrent neural networks [20]. The dataset for this dynamic task of EiM consisted of extracted audio features and emotional annotations. The features and the annotations were for frames of audio of length 500 ms. The best-reported results for this task were obtained using an ensemble of six long short-term memory (LSTM) RNNs with different input sequence lengths. The final output was predicted from the ensemble using an extreme learning machine. This method achieved a root-mean-square error (RMSE) of 0.230 for arousal and 0.303 for valence [21].

[*] Equally contributing authors in this paper.

The research leading to these results has received funding from the European Research Council under the European Unions H2020 Framework Programme through ERC Grant Agreement 637422 EVERYSOUND. Part of the computations leading to these results were performed on a TITAN-X GPU donated by NVIDIA. The authors also wish to acknowledge CSC-IT Center for Science, Finland, for computational resources.

In this paper, we propose a method that exploits the combined modeling capacities of the CNN, RNN, and fully connected (FC) network. We call this the stacked convolutional and recurrent neural network (CRNN). Similar architecture of stacked CNN, RNN, and FC has been used in literature and shown to consistently outperform the previous network configurations for sound event detection [22], speech recognition [23], and music classification [24]. We approach the problem as a regression one and evaluate the method using the EiM dynamic task dataset. We use RMSE as the metric in order to provide a direct comparison with results from existing works. Our proposed method employs only a small fraction of parameters compared to the recent state-of-the-art DBLSTM-based system by Li at al. [25] and performs with significantly better results (i.e. lower RMSE for both arousal and valence recognition). Additionally, we evaluate our method with a raw feature set (log mel-band energy). This is motivated from the fact that neural networks can by themselves learn the first order derivatives and statistics existing in the feature set provided in EiM dataset. These raw features are extracted from the same songs that were utilized in the EiM dataset. The results using the raw feature set are comparable to the ones achieved with the Li et al. system.

The rest of the paper is organized as follows. In Section 2 we present our proposed method. Section 3 describes the dataset, features, metric, baseline and the evaluation procedure. We analyze and discuss the obtained results in Section 4. Finally concluding the paper in Section 5.

2. PROPOSED METHOD

The input to the proposed method is a sequence of audio features extracted from audio. We evaluate the method with two separate audio features - baseline (Section 3.2.1) and raw (Section 3.2.2). Our proposed method, illustrated in Figure 1 is a neural network architecture obtained by stacking a CNN with two branches of FC and RNN one each for arousal and valence. This combined CRNN architecture maps the input audio features into their respective arousal and valence values. The output of the method, arousal and valence values, is in the range of [-1, 1].

In our method, the local shift-invariant features are extracted from the audio features using a CNN with a receptive filter size of 3 × 3. The feature map of the CNN acts as the input to two parallel but identical branches, one for arousal and the other one for valence. Each of these branches consists of the FC, the bidirectional gated recurrent unit (GRU) [26], and the output layer consisting of one node of the maxout layer [27]. Both FC and maxout had their weights shared across time steps. The FC was utilized to reduce the dimensionality of the feature maps from the CNN and, consequently, reduce the number of parameters required by the GRU. The bidirectional GRU layer was utilized to learn the temporal information in the features. The maxout layer was employed as the regression layer due to its ability to approximate a convex, piecewise linear activation function [27].

We use the rectified linear unit (ReLU) [28] activation and the batch normalization [29] for the CNN layer. The FC uses the linear activation and the GRU's use the tanh activation. In the bidirectional GRU, we concatenate together the forward and backward activations. The network was trained using the backpropagation through time alorithm [30], the Adam optimizer with the default parameters in [31], and the RMSE as the loss function. In order to reduce overfitting of the network to training data we use dropout [32] for the CNN and RNN layers and the ElasticNet [33] regularization for the weights and activity of the CNN. The network was implemented using the Keras framework [34] and the Theano backend [35].

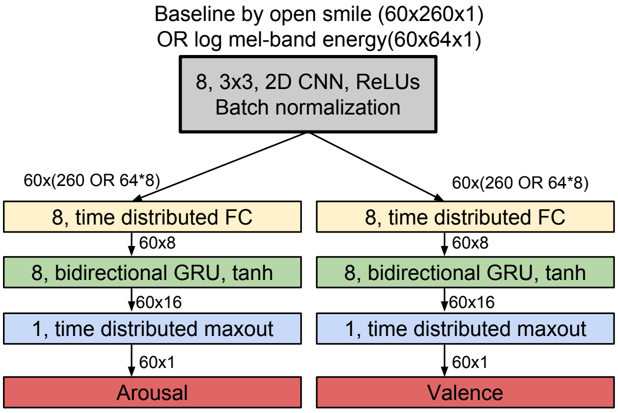

Baseline by open smile (60x260x1)
OR log mel-band energy(60x64x1)
8, 3x3, 2D CNN, ReLUs
Batch normalization
60x(260 OR 64*8)
60x(260 OR 64*8)
8, time distributed FC
8, time distributed FC
60x8
60x8
8, bidirectional GRU, tanh
8, bidirectional GRU, tanh
60x16
60x16
1, time distributed maxout
1, time distributed maxout
60x1
60x1
Arousal
Valence

Figure 1: Proposed method of stacked convolutional and recurrent neural network for music emotion recognition.

3. EVALUATION

3.1 Dataset

For the evaluation of our method, we utilized the dataset provided for the MediaEval's EiM task [15]. For annotations, the Russell's two-dimensional continuous emotional space [36] was used. The values of arousal and valence belonging to every 500 ms segment were annotated by five to seven annotators per song from the Amazon Mechanical Turk.

The training set consisted of 431 audio excerpts with a length of 45 seconds from the Free music archive. The first 15 seconds of excerpts were used for annotator's adaptation. The annotations and features for the remaining 30 seconds were provided as the development set. This amounts to 60 annotations per audio file (30 s of audio with annotations every 500 ms). Each of the annotations are in the range of [-1, 1] for both arousal and valence. Negative values represents low arousal/valence and positive high.

The evaluation set consists of 58 full songs from the royalty free multitrack MedleyDB dataset and Jamendo music website. Similar features and annotations as development set were provided for the evaluation set.

3.2 Audio features

3.2.1 Baseline features

The feature set that we used is the one that was utilized in the EiM challenge. The most of research is based on this feature set, so we call it the baseline features set. It consists of the mean and standard deviation of 65 low-level acoustic descriptors and their first order derivatives from the 2013 INTERSPEECH Computational Paralinguistic Challenge [37], extracted with the openSMILE toolbox [38]. In the feature set are included mel frequency cepstral coefficients (MFCCs), spectral features such as flux, centroid, kurtosis, and rolloff, and voice related features such as jitter and shimmer. The total amount of features is 260 and they were extracted from non overlapping segments of length 500 ms. The mean and standard deviation of the features for this 500 ms were estimated from frames of 60 ms with 50 ms ($\approx 80\%$) overlap.

3.2.2 *Raw audio feature*

The above baseline features consist of mean, standard deviation, first and second order derivatives of different raw features. Neural networks by themselves can extract such statistics from the raw features directly. Thus, we argue that these features are not the best choice for neural networks. In order to study the performance of the proposed network with raw features, we employ just the log mel-band energies. Mel-band related features have been used previously for MediaEval EiM task [18] and MIREX AMC task [39]. The librosa python library [40] was used to extract the mel-band features from 500 ms segments, in a similar fashion to the baseline feature set.

3.3 Metric

The RMSE is widely used in many areas as a statistical metric to measure a model performance. It represents a standard deviation of the differences of predicted values from the line of best fit at the sample level. Given N predicted samples $\hat{y}_n$ and the corresponding reference samples y_n, then RMSE between them can be written as:

$$RMSE = \sqrt{\frac{\sum_{n=1}^{N}(\hat{y}_n - y_n)^2}{N}} \qquad (1)$$

3.4 Baseline

The proposed method is compared to the system proposed in [25]. To the best of our knowledge, this method has the top results on the MediaEval's EiM dataset using the baseline audio features. This result was obtained using an ensemble of six bidirectional LSTM (BLSTM) networks of five layers and 256 units each, trained on sequence length of 10. The first two layers of these were pre-trained with the baseline features. The six networks of the ensemble were chosen such that they covered all the training data, and sequence lengths of 10, 20, 30 and 60. The output of the ensemble of networks was fused using SVR with a third order polynomial kernel and artificial neural network (ANN) of 14 nodes. The average output of SVR and ANN was used as the final arousal and valence values.

In terms of complexity of this baseline method, a five layer 256 input and output units BLSTM has about 2 million parameters. An ensemble of six BLSTMs compounded in this manner will have about 12 million parameters. Additionally, SVR and ANN adds to further complexity of the overall method.

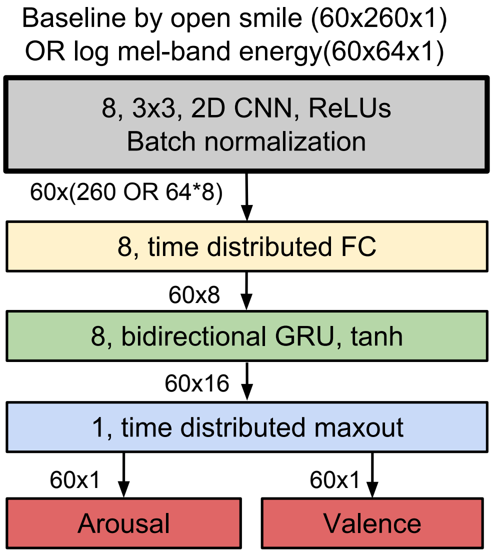

Baseline by open smile (60x260x1)
OR log mel-band energy(60x64x1)
8, 3x3, 2D CNN, ReLUs
Batch normalization
60x(260 OR 64*8)
8, time distributed FC
60x8
8, bidirectional GRU, tanh
60x16
1, time distributed maxout
60x1
60x1
Arousal
Valence

Figure 2: Stacked convolutional and recurrent neural network without branching (CRNN_NB).

3.5 Evaluation procedure

The hyperparameter estimation of the proposed method was done by varying the number of layers of the CNN, FC, and GRU from one to three, and the number of units for each of these were varied in the set of $\{4, 8, 16, 32\}$. Identical dropout rates were used for CNN and GRU and varied in the set of $\{0.25, 0.5, 0.75\}$. The ElasticNet variables L1 and L2 were each varied in the set of {1, 0.1, 0.01, 0.001, 0.0001}. The parameters were decided based on the best RMSE score on the development set, using the baseline features, mini-batch size of 32, and the maximal sequence length of 60. The mini-batch size of 32 was chosen from the set of $\{16, 32, 64, 128\}$ based on the variance in the training set error and the number of iterations taken to achieve the best RMSE. The sequence length was chosen to be the same length as the number of annotations per audio file in the training set, therefore 30 s per every audio clip annotated every 0.5 s and this sequence length varied from 10 s to 60 s with the multiply of 2. The network was trained to achieve the lowest average RMSE of valence and arousal. The best configuration with least number of parameters had one layer each of CNN, FC and GRU with eight units each, a dropout rate of 0.25, L1 of 0.1 and L2 of 0.001. Figure 1 shows the configuration and the feature map dimensions of the proposed method.

In order to provide the comparison of the performance of CRNN without these two branches (CRNN_NB), we removed one branch in the above configuration and trained the CRNN with both the valence and arousal on the same branch as seen in Figure 2.

The proposed CRNN method was trained with the raw features (log mel-band energy). In order to give a direct comparison of performance, we keep the exact configuration as the proposed CRNN with only the dropout changed to 0.75. The network was seen to overfit to the training

Table 1: RMSE errors for development and evaluation data using (a) baseline and (b) log mel-band energy features. The numbers reported are the mean and standard deviation over five separate runs of the method with different initialization of network weights.. The proposed CRNN network performance is reported for sequence lengths 10, 20, 30 and 60.

(a) Baseline features

	Evaluation			Development		
Seq. length	Valence	Arousal	Average	Valence	Arousal	Average
60	0.275±0.004	0.210±0.004	0.242±0.002	0.065±0.002	0.055±0.001	0.060±0.001
30	0.268±0.004	0.205±0.004	0.236±0.001	0.060±0.001	0.053±0.002	0.057±0.001
20	0.268±0.003	**0.202±0.007**	**0.235±0.002**	**0.059±0.001**	**0.050±0.001**	**0.055±0.001**
10	**0.267±0.003**	0.203±0.006	**0.235±0.003**	**0.059±0.002**	0.052±0.001	**0.055±0.001**

(b) Log mel-band energy feature

	Evaluation			Development		
Seq. length	Valence	Arousal	Average	Valence	Arousal	Average
60	0.276±0.006	0.252±0.009	0.264±0.001	0.083±0.002	0.073±0.002	0.078±0.001
30	**0.270±0.003**	0.250±0.004	0.260±0.001	0.079±0.001	0.070±0.001	0.074±0.001
20	**0.270±0.003**	0.248±0.004	0.259±0.002	0.079±0.001	0.069±0.001	0.074±0.000
10	0.273±0.008	**0.244±0.006**	**0.258±0.002**	**0.078±0.000**	**0.067±0.001**	**0.072±0.000**

data with 0.25 dropout rate that used for baseline features. A dropout rate of 0.75 was chosen after tuning in the set of $\{0.25, 0.5, 0.75\}$.

4. RESULTS AND DISCUSSION

Tables 1a and 1b present the RMSE scores on the development and evaluation datasets for the baseline and log mel-band features, respectively. These scores are the mean and standard deviation over five separate runs of the method with different initialization of network weights.

The proposed CRNN method with baseline features gave an average RMSE of 0.242 on evaluation set (see average RMSE for sequence length 60 in Table 1a). The Li et al. baseline method gave an average RMSE of 0.255 (see Table 2). In comparison, the CRNN method has only about 30 k parameters (about 400 times fewer) and fares significantly better in RMSE sense than the Li et al. system. Potentially, further improvement can be achieved by using an ensemble of the proposed method covering different sequence lengths and all training data as proposed in the Li et al. system.

The configuration of the CRNN_NB method had about 17 k parameters and gave the same average RMSE as the Li et al. system (see CRNN_NB in Table 2). This shows the robustness of the proposed method for the emotion recognition task.

Different sequence lengths were experimented with the proposed CRNN, and the results are presented in Table 1. We use a maximum length of 60 which is equal to the number of annotations from a single file in development set, and its factors of 10, 20 and 30. We see that lower sequence lengths of 10 and 20 perform better than using the full-length sequence of 60. Similar observation was reported in [25]. The best average RMSE of 0.235 was achieved with sequences 10 and 20, this is the best result achieved in this paper and is 0.02 less than Li et al.

Table 2: RMSE errors for evaluation data using baseline features. The numbers reported for CRNN_NB are the mean and standard deviation over five separate runs of the method with different initialization of network weights.

Method	Valence	Arousal	Average
Li et al. [25]	0.285	0.225	0.255
CRNN_NB	0.279±0.004	0.231±0.003	0.255±0.002

We compare the performance of the proposed CRNN with log mel-band features in Table 1b. Due to the reduced number of features, this network has only about 10 k parameters. With a simple and not so finely tuned network and raw features we get a best average RMSE of 0.258 with sequence length 10. This is very close to the Li et al. system performance with 1200 times fewer parameters. This proves our hypothesis that the neural network can learn the information of first and second order derivatives and first order statistics from the raw features on its own. The trend of shorter sequences performing better than longer sequences is also observed with log mel-band energy. A network tuned for log mel-band features specifically has a potential to perform better than the Li et al. system.

5. CONCLUSION

In this paper, we proposed a method consisting of stacked convolutional and recurrent neural networks for continuous prediction of emotion in two-dimensional V-A space. The proposed method used significantly less amount of parameters than the Li et al. system and the obtained results outperform those of the Li et al. system.

The proposed CRNN was evaluated with different lengths of sequences, and the smaller sequence lengths were seen to perform better than the longer lengths. Log mel-band energy feature was proposed in place of baseline features, and the proposed CRNN was seen to learn information equivalent to that of baseline features from just the mel-band features.

6. REFERENCES

[1] P. N. Juslin and K. R. Scherer, *Vocal expression of affect.* Oxford University Press, 2005, pp. 65–135.

[2] P. N. Juslin and P. Laukka, “Communication of emotions in vocal expression and music performance: different channels, same code?” *Psychological bulletin*, vol. 129, no. 5, pp. 770–814, Sep 2003.

[3] ——, “Expression, perception, and induction of musical emotions: A review and a questionnaire study of everyday listening,” *Journal of New Music Research*, vol. 33, no. 3, pp. 217–238, 2004.

[4] S. Zhang, Q. Tian, S. Jiang, Q. Huang, and W. Gao, “Affective MTV analysis based on arousal and valence features,” in *2008 IEEE International Conference on Multimedia and Expo*, June 2008, pp. 1369–1372.

[5] T. Li and M. Ogihara, “Content-based music similarity search and emotion detection,” in *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 5, May 2004, pp. V–705–8 vol.5.

[6] P. Ekman, “An argument for basic emotions,” *Cognition and Emotion*, vol. 6, pp. 169–200, May 1992.

[7] K. Hevner, “Experimental studies of the elements of expression in music,” *The American Journal of Psychology*, vol. 48, pp. 246–268, 1936.

[8] K. Drossos, R. Kotsakis, G. Kalliris, and A. Floros, “Sound events and emotions: Investigating the relation of rhythmic characteristics and arousal,” in *IISA 2013*, July 2013, pp. 1–6.

[9] K. Drossos, A. Floros, and N.-G. Kanellopoulos, “Affective acoustic ecology: Towards emotionally enhanced sound events,” in *Proceedings of the 7th Audio Mostly Conference: A Conference on Interaction with Sound*, ser. AM ’12. New York, NY, USA: ACM, 2012, pp. 109–116. [Online]. Available: http://doi.acm.org/10.1145/2371456.2371474

[10] X. Downie, C. Laurier, and M. Ehmann, “The 2007 MIREX audio mood classification task: Lessons learned,” in *Proc. 9th Int. Conf. Music Inf. Retrieval*, 2008, pp. 462–467.

[11] “MIREX 2016: Audio train/test: Mood classification - MIREX08 dataset,” accessed 14.2.2017. [Online]. Available: http://www.music-ir.org/nema_out/mirex2016/results/act/mood_report/summary.html

[12] “MIREX 2011: MIREX 2010: Audio train/test: Mood classification - MIREX08 dataset,” accessed 14.2.2017. [Online]. Available: http://www.music-ir.org/nema_out/mirex2011/results/act/mood_report/summary.html

[13] K. Drossos, A. Floros, A. Giannakoulopoulos, and N. Kanellopoulos, “Investigating the impact of sound angular position on the listener affective state,” *IEEE Transactions on Affective Computing*, vol. 6, no. 1, pp. 27–42, Jan 2015.

[14] H. Xianyu, X. Li, W. Chen, F. Meng, J. Tian, M. Xu, and L. Cai, “SVR based double-scale regression for dynamic emotion prediction in music,” in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on.* IEEE, 2016, pp. 549–553.

[15] A. Aljanaki, Y.-H. Yang, and M. Soleymani, “Emotion in music task at MediaEval 2015.” in *MediaEval*, 2015.

[16] M. Chmulik, I. Guoth, M. Malik, and R. Jarina, “UNIZA system for the” emotion in music” task at mediaeval 2015.” in *MediaEval*, 2015.

[17] V. Imbrasaite and P. Robinson, “Music emotion tracking with continuous conditional neural fields and relative representation.” in *MediaEval*, 2014.

[18] K. Cai, W. Yang, Y. Cheng, D. Yang, and X. Chen, “PKU-AIPL’solution for MediaEval 2015 emotion in music task.” in *MediaEval*, 2015.

[19] B. G. Patra, P. Maitra, D. Das, and S. Bandyopadhyay, “MediaEval 2015: Music emotion recognition based on feed-forward neural network.” in *MediaEval*, 2015.

[20] T. Pellegrini and V. Barrière, “Time-continuous estimation of emotion in music with recurrent neural networks,” in *MediaEval 2015 Multimedia Benchmark Workshop (MediaEval 2015)*, 2015, pp. 1–3.

[21] M. Xu, X. Li, H. Xianyu, J. Tian, F. Meng, and W. Chen, “Multi-Scale approaches to the MediaEval 2015” emotion in music” task.” in *MediaEval*, 2015.

[22] S. Adavanne, P. Pertilä, and T. Virtanen, “Sound event detection using spatial features and convolutional recurrent neural network,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

[23] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, “Convolutional, long short-term memory, fully connected deep neural networks,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

[24] K. Choi, G. Fazekas, M. Sandler, and K. Cho, “Convolutional recurrent neural networks for music classification,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

[25] X. Li, J. Tian, M. Xu, Y. Ning, and L. Cai, “DBLSTM-based multi-scale fusion for dynamic emotion prediction in music,” in *Multimedia and Expo (ICME), 2016 IEEE International Conference on.* IEEE, 2016, pp. 1–6.

[26] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using RNN encoder-decoder for statistical machine translation,” *arXiv preprint arXiv:1406.1078*, 2014.

[27] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio, "Maxout networks," in *International Conference on Machine Learning (ICML)*, 2013.

[28] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[29] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *CoRR*, vol. abs/1502.03167, 2015.

[30] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[31] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *arXiv:1412.6980 [cs.LG]*, 2014.

[32] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," in *Journal of Machine Learning Research (JMLR)*, 2014.

[33] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," in *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 67, no. 2, 2005, pp. 301–320.

[34] F. Chollet, "Keras," https://github.com/fchollet/keras, 2015, accessed 16.01.2017.

[35] Theano Development Team, "Theano: A Python framework for fast computation of mathematical expressions," vol. abs/1605.02688, May 2016. [Online]. Available: http://arxiv.org/abs/1605.02688

[36] J. A. Russell, "A circumplex model of affect," *Journal of Personality and Social Psychology*, vol. 39, no. 6, pp. 1161–1178, 1980.

[37] F. Weninger, F. Eyben, B. W. Schuller, M. Mortillaro, and K. R. Scherer, "On the acoustics of emotion in audio: what speech, music, and sound have in common," *Frontiers in Psychology*, vol. 4, p. 292, 2013. [Online]. Available: http://journal.frontiersin.org/article/10.3389/fpsyg.2013.00292

[38] F. Eyben, F. Weninger, F. Gross, and B. Schuller, "Recent developments in opensmile, the munich open-source multimedia feature extractor," in *Proceedings of the 21st ACM international conference on Multimedia*. ACM, 2013, pp. 835–838.

[39] T. Lidy and A. Schindler, "Parallel convolutional neural networks for music genre and mood classification," *MIREX2016*, 2016.

[40] B. McFee, M. McVicar, C. Raffel, D. Liang, O. Nieto, E. Battenberg, J. Moore, D. Ellis, R. Yamamoto, R. Bittner, D. Repetto, P. Viktorin, J. F. Santos, and A. Holovaty, "librosa: 0.4.1," accessed 16.01.2017. [Online]. Available: http://dx.doi.org/10.5281/zenodo.32193

VALIDATING ATTACK PHASE DESCRIPTORS OBTAINED BY THE TIMBRE TOOLBOX AND MIRTOOLBOX

Kristian Nymoen
Department of Musicology
University of Oslo, Norway
kristian.nymoen@imv.uio.no

Anne Danielsen
Department of Musicology
University of Oslo, Norway
anne.danielsen@imv.uio.no

Justin London
Department of Music
Carleton College, USA
jlondon@carleton.edu

ABSTRACT

The attack phase of sound events plays an important role in how sounds and music are perceived. Several approaches have been suggested for locating salient time points and critical time spans within the attack portion of a sound, and some have been made widely accessible to the research community in toolboxes for Matlab. While some work exists where proposed audio descriptors are grounded in listening tests, the approaches used in two of the most popular toolboxes for musical analysis have not been thoroughly compared against perceptual results. This article evaluates the calculation of attack phase descriptors in the Timbre toolbox and the MIRtoolbox by comparing their predictions to empirical results from a listening test. The results show that the default parameters in both toolboxes give inaccurate predictions for the sound stimuli in our experiment. We apply a grid search algorithm to obtain alternative parameter settings for these toolboxes that align their estimations with our empirical results.

1. INTRODUCTION

Many music researchers rely on available toolboxes for quantitative descriptions of the characteristics of sound files. These descriptions are commonly referred to as *descriptors* or *features*, and they may be used as variables in experimental research or as input to a classification algorithm. In this paper we investigate the descriptors that concern the first part of sonic events, what we will refer to as *attack phase descriptors*.[1] Particularly, we are interested in the detection of salient time points, such as the point when the sound event is first perceived, the point when it reaches its peak amplitude, and the *perceptual attack* of the sound event, which will be introduced properly in Section 2. Robust detection of such points in time is essential to obtain accurate values for the attack phase descriptors commonly used in the music information retrieval community, such as Log-attack time and attack slope. We compare the results of two popular toolboxes for Matlab, the MIRtoolbox [1] (version 1.6.2) and the Timbre Toolbox [2] (version 1.4), in estimating these salient moments. Sound samples from 'real' music recordings are used to compare the toolbox analysis results to a listening test. As such, our research complements the work by Kazazis et al. [3], where the toolbox results are compared to strictly controlled synthesis parameters using additive synthesis.

In Section 2 we discuss previous work in auditory perception concerned with the attack portion of sonic events. In Section 3 we take a closer look at computational estimation of attack phase descriptors. Further, in Section 4, we show how the algorithms in the MIRtoolbox and Timbre toolbox compare to our own experimental results, and then we discuss these results and conclude in Section 5.

2. BACKGROUND

The attack of musical events has been studied from a range of perspectives. Pierre Schaeffer experimented with tape recordings of sounds in the 1960s [4]. By cutting off the beginning of sound events he showed the importance of the attack portion for the perception of sonic events.

A seminal paper in this field is John W. Gordon's study from 1987 of the perceptual attack time of musical tones [5]. Gordon manipulated synthesis parameters in a digital synthesizer and observed how the perceived moment of metrical alignment was affected. A range of parameters were found to be involved. Gordon introduced the term *perceptual attack time*, to describe the moment in time perceived as the rhythmic emphasis of a sound [5]. The term was further explained by Wright, saying that the perceptual attack time of a musical event is its "perceived moment of rhythmic placement" [6]. This definition emphasises that the perceptual attack time of a sound event acts as reference when aligning it with other events to create a perceptually isochronous sequence, as illustrated in Figure 1. Wright further argues that the perceptual attack of a sound event is best modelled not as a single point in time, but as a continuous probability density function indicating the likelihood as to where a listener would locate the maxima of rhythmic emphasis. These definitions are strongly linked to the concept of perceptual centres (P-centres) in speech timing, defining the moment in time when a brief event is perceived to occur [7]. In effect, when two sounds are perceived as synchronous, it is their P-centres that are aligned.

In addition to the perceptual attack time, Gordon [5] argues that it makes sense to talk about both *physical onset time* and *perceptual onset time* for a given sound event.

[1] Note that *phase* in this context signifies that these descriptors concern a certain temporal segment of the sound. This differs from the technical meaning of *phase* in signal processing, such as in a *phase spectrum*.

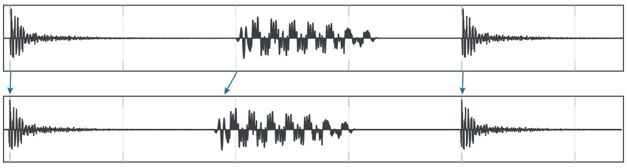


Figure 1. The top plot shows a sequence of sounds arranged isochronously by *physical onset*. The bottom plot shows the sounds arranged isochronously by *perceptual attack*. The latter will be perceptually isochronous.

The former refers to the moment in time when the sound event begins (before it is perceived), and the latter to the moment in time when the sound event is first perceived.

Attack phase descriptors may represent physical features of the sound signal, or be estimations of perceptual features. However, the distinction between the two types is rarely made clear. This may be due to the wide application of audio descriptors in various machine learning tasks, where physical features are sufficient to obtain the required result (e.g. in automatic classification of instrument type [8]). Another related machine learning task is the annual MIREX[2] competition, where the term *onset time* is used to denote the (approximate) time points in an audio file where new sound events occur, without addressing the distinction between physical and perceptual onset.[3] In studies of auditory perception, we argue that it is imperative to be aware of the distinction between the two and the inherent difficulty of estimating perceptual features [9].

2.1 Terminology

Table 1 and Figure 2 show a compilation of attack phase descriptors, most of them found in the Timbre Toolbox and MIRtoolbox. In our use of the term, attack phase descriptors include salient time points within or close to the attack portion of a sound event, in addition to relative measures, such as the time difference between two salient time points.

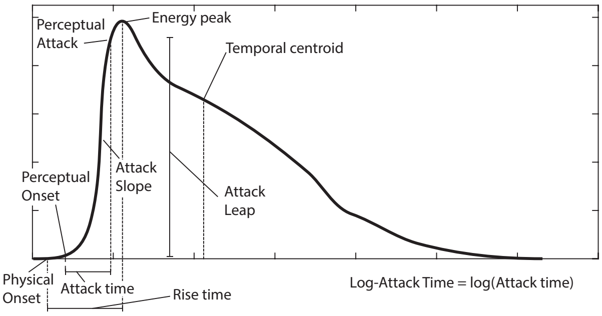


Figure 2. Illustration of various attack phase descriptors.

Name	Type	Description
Physical onset	phTP	Time point where the sound energy first rises from 0.
Perceptual onset	peTP	Time point when the sound event becomes audible.
Perceptual attack	peTP	Time point perceived as the rhythmic emphasis of the sound — what Gordon calls perceptual attack *time* [5].
Energy peak	phTP	Time point when the energy envelope reaches its maximum value.
Rise time	phTS	Time span between physical onset and energy peak.
Attack time	peTS	Time span between perceptual onset and perceptual attack.
Log-Attack Time	phTS	The base 10 logarithm of attack time.
Attack slope	peES	Weighted average of the energy envelope slope in the attack region [2].
Attack leap	peES	The difference between energy level at perceptual attack and perceptual onset [10].
Temporal centroid	phTP	The temporal barycentre of the sound event's energy envelope.

Table 1. Table of attack phase descriptors. ph: physical, pe: perceptual, T: time, E: energy, S: span, P: point.

The "Type"-column in Table 1 refers to particular characteristics of the descriptor. A descriptor marked 'ph' describes a physical aspect of the audio signal, and 'pe' describes perceptual aspects of the sound event. What we refer to as time point descriptors (TP) indicate a single point in time. Note that we choose not to use the word 'time' in TP descriptors, to emphasise that these are conceptually different from descriptors representing a time span. As noted by Wright [6] the reference of such time points may be a fixed point in the sound file (e.g., beginning), or an external time reference (e.g., a SMPTE clock). Further, the time points may be understood in reference to some other calculated time point, for instance the perceptual attack in relation to the physical onset. Gordon [5] used this measure, naming it *relative perceptual attack time*. It is worth mentioning that detection of physical onset may be unreliable. For natural sounds and recordings, the estimation of physical onset will depend on algorithmic variables; typically, a noise threshold to eliminate background noise, and some parameters as input to the envelope extraction algorithm. For digitally synthesized audio signals, however, the first nonzero signal value can be detected and the physical onset can confidently be used as a reference.

Time span descriptors (TS) describe the temporal relation between two time point descriptors. In effect, this category is not mutually exclusive to the TP category: a TP descriptor with a local time reference (e.g. the time of the physical onset of the sound event) can also be seen as a TS descriptor, such as Gordon's relative perceptual attack time.

We use the term energy span descriptors (ES) to denote descriptors that describe how the energy envelope develops between two time points. Finally, although not shown in Table 1, one could easily include EP descriptors to denote the energy level at salient time points.

[2] `http://www.music-ir.org/mirex`

[3] The goal in the MIREX audio onset competitions is to detect discrete sound events in an audio file that has been annotated by a group of music experts. False positives (indicating a new sound event where an expert annotator did not) and false negatives (missing a sound event where an expert annotator indicated one) are penalised. In other words, high temporal precision for the extracted events is not a primary goal in the MIREX audio onset competition, which explains why the distinction between physical and perceptual onset is not addressed.

3. COMPUTING ATTACK PHASE DESCRIPTORS

In this section we look more closely at how various attack phase descriptors are computed in the Timbre Toolbox and MIRtoolbox. With reference to the list of descriptors in Table 1, the computational estimation of *ph*-type descriptors is conceptually quite different from that of *pe*-type descriptors. While a physical feature of the signal can be calculated accurately, and only depends on algorithmic parameters like filter cutoff frequency or window length, any perceptual feature can only be an estimate of how the particular sound event would be perceived. Neither of the toolboxes state clearly whether the computed descriptors are estimates of perceptual or physical features. Both employ algorithms where the end of the attack range does not necessarily coincide with the peak of the energy envelope. This suggests that the toolboxes take into account the fact that the perceived attack of a sound event might end before the energy peak.

Below, we take a closer look at the attack phase descriptors as they are provided by the toolboxes, and in Section 4 we compare their output to perceptual results from our own listening experiment. Conceptually, the two toolboxes take similar approaches to computing attack phase descriptors. However, on an algorithmic level, the two are quite different. For now, we consider the default strategies and parameters in each toolbox and look at how the energy envelope and attack phase descriptors are calculated.

3.1 Energy envelope

The common basis of most algorithms for calculating attack phase descriptors is some function describing the *energy envelope* of the sound signal. Attack phase descriptors are calculated based on various thresholds applied to this envelope and its derivatives. The method and the parameters specified in the calculation of the energy envelope strongly influence the estimated attack phase descriptors.

The default strategy in the Timbre Toolbox is to apply a Hilbert transform to the audio signal, followed by a 3rd-order Butterworth lowpass filter with cutoff frequency at 5 Hz [2]. The Timbre Toolbox does not compensate for group delay in the filter when extracting the energy envelope. This is not crucial to the extraction of TS attack phase descriptors such as Log-Attack Time, but delays all TP descriptors by the group delay time of the filter.[4]

The MIRtoolbox implements a range of strategies, and the default strategy in onset detection with attack estimation[5] is to calculate the envelope as the sum of columns in a spectrogram using a hanning window of 100 ms with hop factor at 10%.

3.2 Extracting attack phase descriptors

The two toolboxes we explore in this paper use different terminology. In the MIRtoolbox the term ‘onset’ is used to describe the sound events extracted by peak detection of the energy envelope. This makes sense when seen in relation to the previously mentioned MIREX audio onset competition, but is distinctly different to the meaning of onset in our paper. Both toolboxes provide estimates of when the attack of a sound event begins and ends.[6] Both also provide several directly or indirectly related descriptors (such as attack slope and temporal centroid).

Peeters et al. [2] argue that while a common approach in estimating the beginning and end of an attack range is to apply fixed thresholds to the energy envelope, more robust results may be obtained by looking at the slope of the energy envelope before it reaches its peak value. Trumpet sounds are mentioned as a particular reason for this, as their energy envelopes may increase after the attack. Both of the toolboxes we discuss take such an approach, however with different strategies. The two strategies are illustrated in Figure 3, and explained further below.

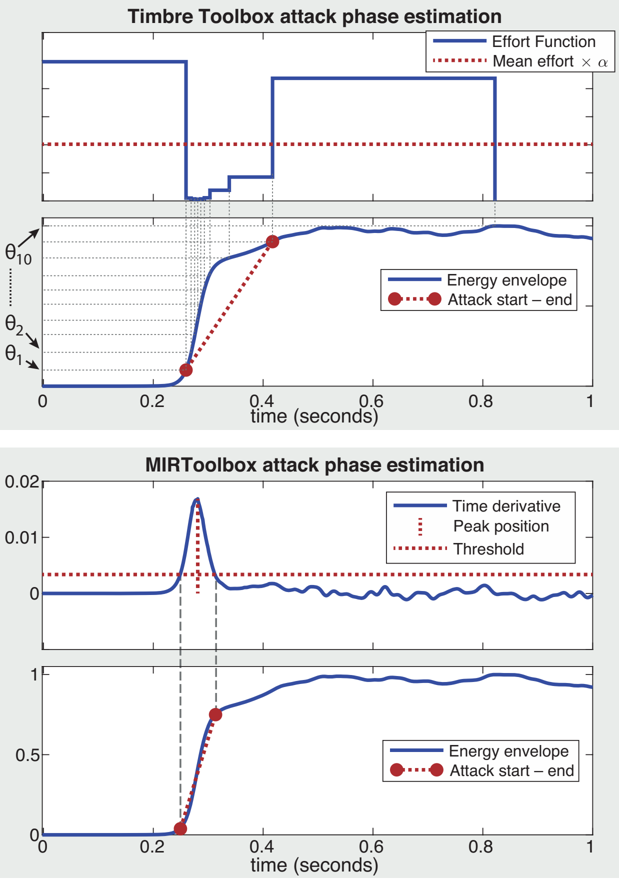


Figure 3. Attack estimation in the Timbre Toolbox and MIRtoolbox. The figure shows the process of estimating the attack start and attack end. The Timbre Toolbox uses an approach that examines the time interval between successive energy thresholds. The MIRtoolbox puts a threshold on the first derivative of the energy envelope. For comparability, this figure uses an identical energy envelope (from a clarinet sound) for both algorithms.

[4] Examples of delayed envelopes are shown Figure 4 in Section 4.2.

[5] This is the default strategy for MIRtoolbox *onset detection*, other functions in the toolbox rely on different calculations of the energy envelope.

[6] In the MIRtoolbox, the attack range is estimated by passing an optional ‘Attacks’-argument to the `mironset()` function. As per version 1.6.2 of the toolbox, the `mirgetdata()` function will return the energy peak rather than the extracted attack region. For an object `A` containing the analysis of an audio file, the required code to obtain the attack start is `uncell(get(A,'AttackPosUnit'))` and to obtain the attack end, the code `uncell(get(A,'PeakPosUnit'))` may be used.

Peeters [11] suggested the *weakest effort method*, which is implemented in the Timbre Toolbox for estimating the beginning and end of the attack. In this method, a set of i equidistant thresholds θ_i is specified between zero and a peak value for the energy envelope. For each θ_i, a corresponding t_i denotes the first time when the energy envelope crosses the corresponding threshold. Then a set of 'effort' parameters $\omega_{i,i+1}$ is calculated as the time between each t. With the mean effort notated as $\overline{\omega}$, the beginning of the attack is calculated as the first $\omega_{i,i+1}$ that falls below $\overline{\omega} \times \alpha$, and the end as the first sucessive ω_i that exceeds $\overline{\omega} \times \alpha$. Both are also adjusted to the local maximum/minimum within the $\omega_{i,i+1}$ interval. The default value for α is 3.

The MIRtoolbox uses the first time derivative of the energy envelope (e') as basis for estimating attack phase descriptors. A peak in e' necessarily occurs before the energy peak itself, and the attack is predicted to begin when e' goes above 20% of its peak value, and end when it falls below 20% of its peak value.

4. ANALYSIS

After the previous look at the toolbox algorithms, we will now compare their calculations to results from a perceptual experiment. First, we describe the experiment, followed by an analysis of attack estimation by the two toolboxes using their default parameter settings. Subsequently, we perform a parameter optimisation on the toolbox algorithms to improve the alignment between the calculated estimations and our perceptual results.

4.1 Experiment

In our experiment 17 participants were asked to align a click track to a set of 9 sound stimuli. Each stimulus was presented repeatedly at an interval of 600 ms, along with a sequence of clicks every 600 ms. For each trial, the click track and stimuli started with a random offset. Participants adjusted the alignment of the clicks to the stimuli using a keyboard and/or a slider on the screen, until they perceived the two streams as synchronous. The task was repeated four times per stimulus. All were given a gift certificate worth NOK 200 (≈ 22 €) for their participation.

Eight isolated instrumental sounds were chosen as stimuli. These were selected with an aim to obtain stimuli with different perceptual characteristics and musical function. The click-sound from the click track was also included as a stimulus. The physical onset of each stimulus was annotated manually through inspection of the waveform and used as reference in the results presented below.

The results from the experiment show that our participants did not agree on an exact point in time where the stimulus and click tracks were aligned. Alignment of the click sound to itself was an exception, with a standard deviation of only 1 ms. The agreement on the location of the perceptual attack varied with the characteristics of the sound stimuli. Not counting the task of aligning the click track to itself, the standard deviations of the time delay between click track and stimulus track ranged between 7 ms (for bright sounds with fast attack and short duration) and 18 ms (dark sounds with slow attack and long duration). This verifies previous research, which suggests that perceived attack may best be modelled as a range (i.e. a 'beat bin' [12]) or probability distribution [6], rather than a single point in time. Consequently, we cannot evaluate the toolboxes based on a single value (e.g., their estimation of the beginning of the attack), but rather need to determine the amount of overlap between the calculated attack range and the distribution of perceptual responses.

The use of a simultaneous click track and stimulus track may provide fusion cues that are not inherent to the perceptual attack of the stimulus. Several scholars have argued that an alternating sequence between stimulus and click might be more reliable [9, 13]. We controlled for the effects of event fusion by a corresponding anti-phase alignment task, alternating click and stimulus. The results showed no significant difference between the two response modes [14], which is also in accordance with Villing's reported concistency across these measurement methods [7].

4.2 Energy envelope of the stimuli

In our analysis, we found that the toolboxes' default settings extract imprecise energy envelopes for certain types of sound. In particular, we observe that too long energy envelopes are calculated for short sounds, and that the energy envelopes fail to identify fast attacks in low-frequency sounds with brief high-frequency onsets (e.g. a bass drum sound). This confirms the finding of Kazazis et al. [3], that attack times for fast attack envelopes are largely overestimated in both toolboxes. As previously mentioned, the Timbre toolbox does not compensate for group delay in the lowpass filter. Consequently, the energy envelopes are not in sync with the waveform, and in the most extreme cases hardly overlap at all. The effect of the group delay can be seen in Figure 4, which shows estimated energy envelopes of a kick drum sound. The figure also shows the smearing that occurs with default parameters (D), and how a tighter fit to the waveform may be obtained with an alternate parameter setting (A) for each toolbox. The alternate settings in this figure correspond to a cutoff frequency of 40 Hz for the lowpass filter in the Timbre toolbox, and for the MIRtoolbox a frame size of 50 ms with 2% hop factor.

For the results presented in the following section, we used Matlab's `filtfilt` function to compensate for the group delay in the Timbre toolbox. These results were marginally better when compared to the version with group delay.

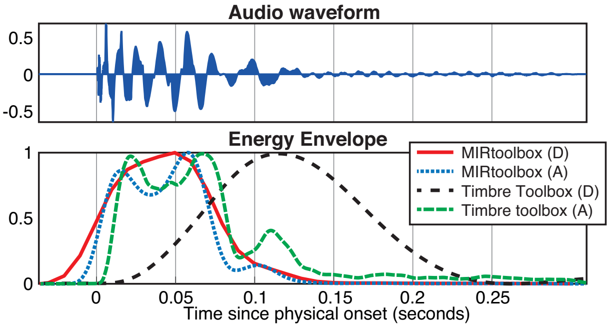


Figure 4. Normalised energy envelopes extracted by the toolboxes. D = default settings, A = alternate settings.

4.3 Attack detection

The toolboxes estimate the attack range using the algorithms presented in Section 3.2. In addition to envelope extraction parameters, each toolbox relies on a predefined threshold, the default values being $\alpha = 3$ for the Timbre Toolbox and 0.2 times e'_{peak} for MIRtoolbox.

We have compared the attack range estimated by the two toolboxes to the time span covered by mean±SD in our perceptual experiment. We acknowledge the problem of comparing two such different measures (one denoting a time range, and the other a distribution of single time points). However, we argue that a good computational estimate of a perceptual attack range should overlap with the majority of responses in the perceptual data.

The results of attack detection in both toolboxes are illustrated in Figure 5. The figure shows results from default parameters and an optimised set of parameters as will be explained in Section 4.4. For each stimulus, the four vertical lines indicate the four different approaches. The boxes and black horizontal lines indicate the results (mean±SD) from our perceptual test. In summary, the default settings for both toolboxes result in quite long onset periods, compared to the range indicated by the results from the perceptual experiment. The mean estimated attack time (duration of attack) for all the sounds were 48 ms (MIRtoolbox) and 96 ms (Timbre toolbox), while the mean interval corresponding to two standard deviations from our perceptual results was 22 ms.

4.4 Optimisation

Although several of the algorithmic parameters are hard-coded (i.e. not intended for user adjustment), one may modify the code in order to run an optimisation algorithm on the parameters. We have used a two-dimensional grid search optimisation on two parameters for each toolbox: one envelope parameter and one threshold parameter, to minimise the difference between the toolbox results and our perceptual data. For each parameter setting, we computed the Jaccard Index [15] between the estimated attack range and the time span covered by our experimental results (mean±SD). This is a measure of the amount of overlap between the two, and takes a value of 1 if the two ranges are identical, and 0 if there is no overlap. For the default parameter settings, the mean Jaccard Index across all sounds were 0.41 (MIRtoolbox) and 0.25 (Timbre toolbox).

Figure 6 shows the output of our grid search algorithm for the MIRtoolbox. The mean Jaccard Index across the nine sounds was used as fitness measure. 30 settings were tested per parameter, in total 900 parameter settings per toolbox. The results from the parameter optimisation process are shown in Table 2, and the corresponding optimised estimates of attack ranges are shown in Figure 5. The Jaccard Index scores for the optimised parameters were 0.57 for the MIRtoolbox and 0.61 for the Timbre toolbox. The main improvement is the reduced spread of the attack times compared with the default attack estimates, as a result of more detailed energy envelopes.

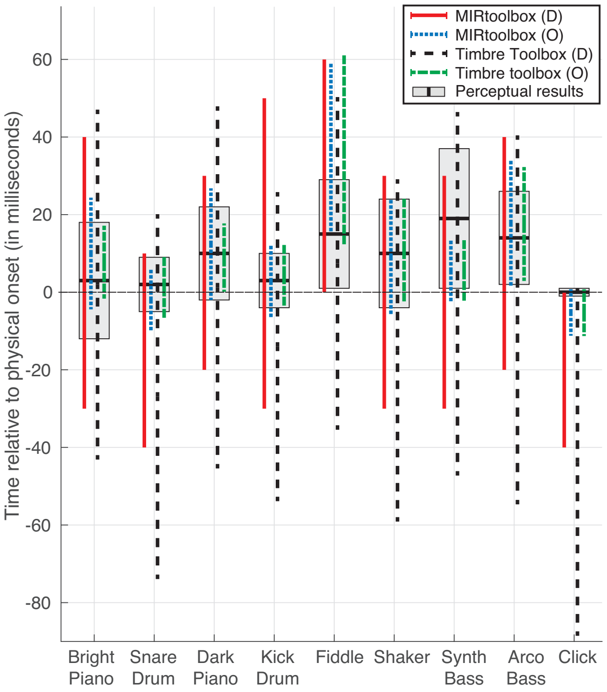


Figure 5. Default (D) and optimised (O) estimates of attack ranges of the 9 stimuli. Boxes show perceptual results for each sound file (mean±SD). Physical onset (time 0) was manually determined by inspecting the waveform.

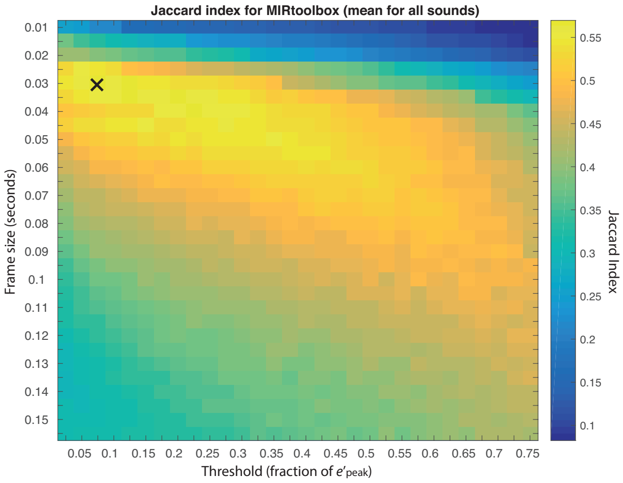


Figure 6. Output of grid search algorithm optimising frame size and threshold parameters in the MIRtoolbox.

Toolbox	Envelope parameter	Threshold parameter
Timbre toolbox	LPfilter cutoff frequency Default: 5 Hz Optimised: 37 Hz	α Default: 3 Optimised: 3.75
MIR-toolbox	Frame size Default: 0.1 s Optimised: 0.03 s	fraction of e'_{peak} Default: 0.2 Optimised: 0.075

Table 2. Parameters for optimisation, default values and optimised results

5. DISCUSSION

Estimating perceptual parameters computationally is difficult. Our results show that attack phase descriptors are no exception. The common first stage in attack detection is extraction of an energy envelope. Both of the toolboxes we have investigated render slowly changing energy envelopes by default. In the MIRtoolbox this comes as result of a large frame size, and in the Timbre toolbox a lowpass filter cutoff frequency of 5 Hz. The default parameters have the advantage of limiting the impact on the energy envelope of rapidly oscillating amplitude modulations in the audible range, in effect reducing the likelihood of false positives in MIREX-style onset detection tasks. However, the low cutoff frequency also hinders accurate representation of fast, non-oscillating amplitude changes (e.g. sounds with a short rise time). Interestingly, the authors of the Timbre toolbox acknowledge the need for a higher cutoff frequency and also address the filter group delay problem in a discussion of sound sample duration [2], but no solution is provided in the toolbox. Furthermore, only one of the four parameters subjected to optimisation in our experiment is easily accessible to the user. The MIRtoolbox allows the user to specify frame size as an input parameter to the `mironsets()` function. The Timbre toolbox cutoff frequency, and the threshold parameters for each toolbox (α and fraction of e'_{peak}) are hard-coded, basically leaving them inaccessible to less experienced users.

As noted by a number of scholars, the perceptual attack of a sound event cannot be measured directly [5–7, 9]. The common approach is to estimate the perceptual attack of a sound by comparing its alignment to some other sound with short duration. This in itself induces some uncertainty into perceptual attack experiments, as the perceptual attack of the reference sound is also unknown. The latter is, however, not a problem in our experiment, given the very precise perceptual results for the test sound (the perceived P-center of the click is 0 and the SD is less than 1 ms). Another complicating factor is that the perceptual attack of most sounds might best be modelled as a range, or a probability distribution, rather than as a single point in time [6]. Consequently, the collection of measured single-point indications of perceptual attack must be considered to represent a time range. We have chosen to represent this range by the by the mean±SD of our perceptual results. Thus, the widths of all ranges depend on this purely statistical measure. Ranges of different widths, with different corresponding sets of parameter optima, would have been obtained if a dispersion measure other than the standard deviation had been chosen. In future research we aim to investigate if Wright's distribution models for attack time could provide better estimates for the perceived width of the attack range [6].

The limited size of our experiment, with only 9 sound stimuli and 17 participants, engenders a chance of overfitting the parameters to our data. Before the results of parameter optimisation can be generalised, a larger corpus of sound files and more perceptual data must be investigated.

Our results show that the attack estimates provided by the two Matlab toolboxes are largely dependent on the input parameters used. Both toolboxes seem by default not to be oriented towards sounds with a fast rise time. With appropriate parameters, however, both toolboxes may provide estimates closer to perceptual results for a wider range of sounds. The toolboxes are in general excellent tools for sonic research, and may also be used where accurate timing of events is essential. However, we advise to take caution against using the default parameters as perceptual estimates and note that one must carefully select the parameters used in estimation of attack phase descriptors.

6. REFERENCES

[1] O. Lartillot and P. Toiviainen, "Mir in matlab (ii): A toolbox for musical feature extraction from audio," in *Proceedings of the Int. Conf. on Music Information Retrieval*, Vienna, 2007.

[2] G. Peeters, B. Giordano, P. Susini, N. Misdariis, and S. McAdams, "The timbre toolbox: Extracting audio descriptors from musical signals," *The Journal of the Acoustical Society of America*, vol. 130, no. 5, pp. 2902–2916, 2011.

[3] S. Kazazis, E. Nicholas, P. Depalle, and S. McAdams, "A performance evaluation of the timbre toolbox and the mirtoolbox on calibrated test sounds," in *Proc. of the Int. Symposium on Musical Acoustics (ISMA)*, Montreal, 2017 (forthcoming).

[4] P. Schaeffer and G. Reibel, *Solfège de l'objet sonore*, INA-GRM 1998 ed. Paris, France: ORTF, 1967.

[5] J. W. Gordon, "The perceptual attack time of musical tones," *J. Acoust. Soc. Am.*, vol. 82, no. 1, pp. 88–105, 1987.

[6] M. Wright, "The shape of an instant: Measuring and modeling perceptual attack time with probability density functions," Ph.D. dissertation, Stanford University, 2008.

[7] R. Villing, "Hearing the moment: Measures and models of the perceptual centre," Ph.D. dissertation, National University of Ireland, 2010.

[8] J. D. Deng, C. Simmermacher, and S. Cranefield, "A study on feature analysis for musical instrument classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 38, no. 2, pp. 429–438, 2008.

[9] N. Collins, "Investigating computational models of perceptual attack time," in *Proceedings of the 9th Int. Conference on Music Perception and Cognition*, 2006, pp. 923–929.

[10] O. Lartillot, "Mirtoolbox 1.6.1 user's manual," Aalborg University, Denmark, Tech. Rep., 2014.

[11] G. Peeters, "A large set of audio features for sound description (similarity and classification) in the CUIDADO project," IRCAM, Tech. Rep., 2004.

[12] A. Danielsen, M. R. Haugen, and A. R. Jensenius, "Moving to the beat: Studying entrainment to micro-rhythmic changes in pulse by motion capture," *Timing & Time Perception*, vol. 3, no. 1-2, pp. 133–154, 2015.

[13] R. Polfreman, "Comparing onset detection and perceptual attack time," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2013.

[14] J. London, K. Nymoen, M. Thompson, D. L. Code, and A. Danielsen, "Where is the beat in that note? comparing methods for identifying the p-center of musical sounds," To appear at the *16th Rhythm Production and Perception Workshop*, Birmingham, UK, 2017.

[15] M. Levandowsky and D. Winther, "Distance between sets," *Nature*, vol. 234, no. 5323, pp. 34–35, 11 1971.

SAMPLE-LEVEL DEEP CONVOLUTIONAL NEURAL NETWORKS FOR MUSIC AUTO-TAGGING USING RAW WAVEFORMS

Jongpil Lee **Jiyoung Park** **Keunhyoung Luke Kim** **Juhan Nam**
Korea Advanced Institute of Science and Technology (KAIST)
[richter, jypark527, dilu, juhannam]@kaist.ac.kr

ABSTRACT

Recently, the end-to-end approach that learns hierarchical representations from raw data using deep convolutional neural networks has been successfully explored in the image, text and speech domains. This approach was applied to musical signals as well but has been not fully explored yet. To this end, we propose sample-level deep convolutional neural networks which learn representations from very small grains of waveforms (e.g. 2 or 3 samples) beyond typical frame-level input representations. Our experiments show how deep architectures with sample-level filters improve the accuracy in music auto-tagging and they provide results comparable to previous state-of-the-art performances for the Magnatagatune dataset and Million Song Dataset. In addition, we visualize filters learned in a sample-level DCNN in each layer to identify hierarchically learned features and show that they are sensitive to log-scaled frequency along layer, such as mel-frequency spectrogram that is widely used in music classification systems.

1. INTRODUCTION

In music information retrieval (MIR) tasks, raw waveforms of music signals are generally converted to a time-frequency representation and used as input to the system. The majority of MIR systems use a log-scaled representation in frequency such as mel-spectrograms and constant-Q transforms and then compress the amplitude with a log scale. The time-frequency representations are often transformed further into more compact forms of audio features depending on the task. All of these processes are designed based on acoustic knowledge or engineering efforts.

Recent advances in deep learning, especially the development of deep convolutional neural networks (DCNN), made it possible to learn the entire hierarchical representations from the raw input data, thereby minimizing the input data processing by hands. This end-to-end hierarchical learning was attempted early in the image domain, particularly since the DCNN achieves break-through results in image classification [1]. These days, the method of stacking small filters (e.g. 3x3) is widely used after it has been found to be effective in learning more complex hierarchical filters while conserving receptive fields [2]. In the text domain, the language model typically consists of two steps: word embedding and word-level learning. While word embedding plays a very important role in language processing [3], it has limitations in that it is learned independently from the system. Recent work using CNNs that take character-level text as input showed that the end-to-end learning approach can yield comparable results to the word-level learning system [4, 5]. In the audio domain, learning from raw audio has been explored mainly in the automatic speech recognition task [6–10]. They reported that the performance can be similar to or even superior to that of the models using spectral-based features as input.

This end-to-end learning approach has been applied to music classification tasks as well [11, 12]. In particular, Dieleman and Schrauwen used raw waveforms as input of CNN models for music auto-tagging task and attempted to achieve comparable results to those using mel-spectrograms as input [11]. Unfortunately, they failed to do so and attributed the result to three reasons. First, their CNN models were not sufficiently expressive (e.g. a small number of layers and filters) to learn the complex structure of polyphonic music. Second, they could not find an appropriate non-linearity function that can replace the log-based amplitude compression in the spectrogram. Third, the bottom layer in the networks takes raw waveforms in frame-level which are typically several hundred samples long. The filters in the bottom layer should learn all possible phase variations of periodic waveforms which are likely to be prevalent in musical signals. The phase variations within a frame (i.e. time shift of periodic waveforms) are actually removed in the spectrogram.

In this paper, we address these issues with sample-level DCNN. What we mean by "sample-level" is that the filter size in the bottom layer may go down to several samples long. We assume that this small granularity is analogous to pixel-level in image or character-level in text. We show the effectiveness of the sample-level DCNN in music auto-tagging task by decreasing strides of the first convolutional layer from frame-level to sample-level and accordingly increasing the depth of layers. Our experiments show that the depth of architecture with sample-level filters is proportional to the accuracy and also the architecture achieves results comparable to previous state-of-the-art performances for the MagnaTagATune dataset and the Million Song Dataset. In addition, we visualize filters learned in the sample-level DCNN.

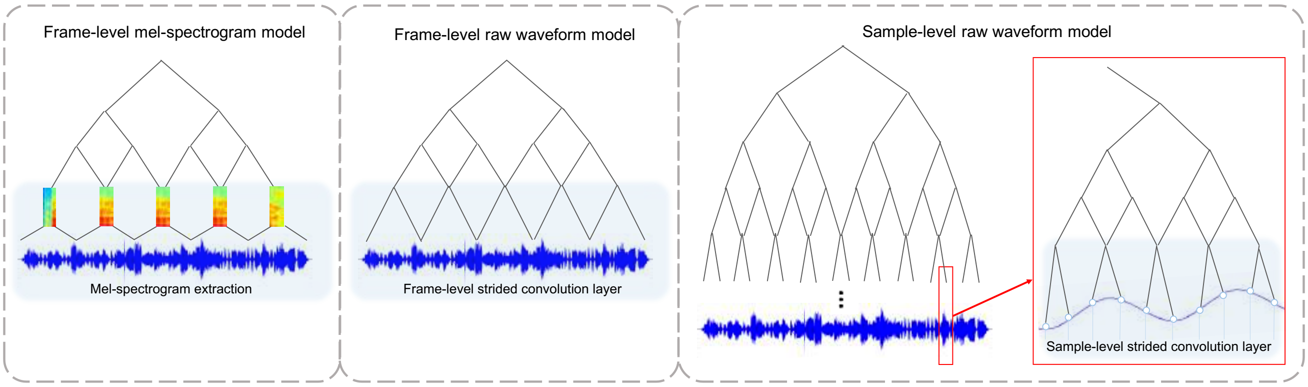

Frame-level mel-spectrogram model
Mel-spectrogram extraction
Frame-level raw waveform model
Frame-level strided convolution layer
Sample-level raw waveform model
Sample-level strided convolution layer

Figure 1. Simplified model comparison of frame-level approach using mel-spectrogram (left), frame-level approach using raw waveforms (middle) and sample-level approach using raw waveforms (right).

2. RELATED WORK

Since audio waveforms are one-dimensional data, previous work that takes waveforms as input used a CNN that consists of one-dimensional convolution and pooling stages. While the convolution operation and filter length in upper layers are usually similar to those used in the image domain, the bottom layer that takes waveform directly conducted a special operation called *strided convolution*, which takes a large filter length and strides it as much as the filter length (or the half). This frame-level approach is comparable to hopping windows with 100% or 50% hop size in a short-time Fourier transform. In many previous works, the stride and filter length of the first convolution layer was set to 10-20 ms (160-320 samples at 16 kHz audio) [8,10–12].

In this paper, we reduce the filter length and stride of the first convolution layer to the sample-level, which can be as small as 2 samples. Accordingly, we increase the depth of layers in the CNN model. There are some works that use 0.6 ms (10 samples at 16 kHz audio) as a stride length [6, 7], but they used a CNN model only with three convolution layers, which is not sufficient to learn the complex structure of musical signals.

3. LEARNING MODELS

Figure 1 illustrates three CNN models in the music auto-tagging task we compare in our experiments. In this section, we describe the three models in detail.

3.1 Frame-level mel-spectrogram model

This is the most common CNN model used in music auto-tagging. Since the time-frequency representation is two dimensional data, previous work regarded it as either two-dimensional images or one-dimensional sequence of vectors [11, 13–15]. We only used one-dimensional(1D) CNN model for experimental comparisons in our work because the performance gap between 1D and 2D models is not significant and 1D model can be directly compared to models using raw waveforms.

3.2 Frame-level raw waveform model

In the frame-level raw waveform model, a strided convolution layer is added beneath the bottom layer of the frame-level mel-spectrogram model. The strided convolution layer is expected to learn a filter-bank representation that correspond to filter kernels in a time-frequency representation. In this model, once the raw waveforms pass through the first strided convolution layer, the output feature map has the same dimensions as the mel-spectrogram. This is because the stride, filter length, and number of filters of the first convolution layer correspond to the hop size, window size, and number of mel-bands in the mel-spectrogram, respectively. This configuration was used for music auto-tagging task in [11, 12] and so we used it as a baseline model.

3.3 Sample-level raw waveform model

As described in Section 1, the approach using the raw waveforms should be able to address log-scale amplitude compression and phase-invariance. Simply adding a strided convolution layer is not sufficient to overcome the problems. To improve this, we add multiple layers beneath the frame-level such that the first convolution layer can handle much smaller length of samples. For example, if the stride of the first convolution layer is reduced from 729 ($= 3^6$) to 243 ($= 3^5$), 3-size convolution layer and max-pooling layer are added to keep the output dimensions in the subsequent convolution layers unchanged. If we repeatedly reduce the stride of the first convolution layer this way, six convolution layers (five pairs of 3-size convolution and max-pooling layer following one 3-size strided convolution layer) will be added (we assume that the temporal dimensionality reduction occurs only through max-pooling and striding while zero-padding is used in convolution to preserve the size). We describe more details on the configuration strategy of sample-level CNN model in the following section.

3.4 Model Design

Since the length of an audio clip is variable in general, the following issues should be considered when configuring the temporal CNN architecture:

- Convolution filter length and sub-sampling length
- The temporal length of hidden layer activations on the last sub-sampling layer
- The segment length of audio that corresponds to the input size of the network

First, we attempted a very small filter length in convolutional layers and sub-sampling length, following the VGG net that uses filters of 3×3 size and max-pooling of 2×2 size [2]. Since we use one-dimensional convolution and sub-sampling for raw waveforms, however, the filter length and pooling length need to be varied. We thus constructed several DCNN models with different filter length and pooling length from 2 to 5, and verified the effect on music auto-tagging performance. As a sub-sampling method, max-pooling is generally used. Although sub-sampling using strided convolution has recently been proposed in a generative model [9], our preliminary test showed that max-pooling was superior to the stride sub-sampling method. In addition, to avoid exhausting model search, a pair of single convolution layer and max-pooling layer with the same size was used as a basic building module of the DCNN.

Second, the temporal length of hidden layer activations on the last sub-sampling layer reflects the temporal compression of the input audio by successive sub-sampling. We set the CNN models such that the temporal length of hidden layer activation is one. By building the models this way, we can significantly reduce the number of parameters between the last sub-sampling layer and the output layer. Also, we can examine the performance only by the depth of layers and the stride of first convolution layer.

Third, in music classification tasks, the input size of the network is an important parameter that determines the classification performance [16, 17]. In the mel-spectrogram model, one song is generally divided into small segments with 1 to 4 seconds. The segments are used as the input for training and the predictions over all segments in one song are averaged in testing. In the models that use raw waveform, the learning ability according to the segment size has been not reported yet and thus we need to examine different input sizes when we configure the CNN models.

Considering all of these issues, we construct m^n-DCNN models where m refers to the filter length and pooling length of intermediate convolution layer modules and n refers to the number of the modules (or depth). An example of m^n-DCNN models is shown in Table 1 where m is 3 and n is 9. According to the definition, the filter length and pooling length of the convolution layer are 3 other than the first strided convolution layer. If the hop size (stride length) of the first strided convolution layer is 3, the time-wise output dimension of the convolution layer becomes 19683 when the input of the network is 59049 samples. We call this “3^9 model with 19683 frames and 59049 samples as input”.

4. EXPERIMENTAL SETUP

In this section, we introduce the datasets used in our experiments and describe experimental settings.

3^9 model, 19683 frames 59049 samples (2678 ms) as input			
layer	stride	output	# of params
conv 3-128	3	19683×128	512
conv 3-128	1	19683×128	49280
maxpool 3	3	6561×128	
conv 3-128	1	6561×128	49280
maxpool 3	3	2187×128	
conv 3-256	1	2187×256	98560
maxpool 3	3	729×256	
conv 3-256	1	729×256	196864
maxpool 3	3	243×256	
conv 3-256	1	243×256	196864
maxpool 3	3	81×256	
conv 3-256	1	81×256	196864
maxpool 3	3	27×256	
conv 3-256	1	27×256	196864
maxpool 3	3	9×256	
conv 3-256	1	9×256	196864
maxpool 3	3	3×256	
conv 3-512	1	3×512	393728
maxpool 3	3	1×512	
conv 1-512	1	1×512	262656
dropout 0.5	–	1×512	
sigmoid	–	50	25650
Total params			1.9×10^6

Table 1. Sample-level CNN configuration. For example, in the layer column, the first 3 of “conv 3-128” is the filter length, 128 is the number of filters, and 3 of “maxpool 3” is the pooling length.

4.1 Datasets

We evaluate the proposed model on two datasets, MagnaTagATune dataset (MTAT) [18] and Million Song Dataset (MSD) annotated with the Last.FM tags [19]. We primarily examined the proposed model on MTAT and then verified the effectiveness of our model on MSD which is much larger than MTAT [1] . We filtered out the tags and used most frequently labeled 50 tags in both datasets, following the previous work [11], [14, 15] [2] . Also, all songs in the two datasets were trimmed to 29.1 second long and resampled to 22050 Hz as needed. We used AUC (Area Under Receiver Operating Characteristic) as a primary evaluation metric for music auto-tagging.

[1] MTAT contains 170 hours long audio and MSD contains 1955 hours long audio in total

[2] `https://github.com/keunwoochoi/MSD_split_for_tagging`

2^n models									
model with 16384 samples (743 ms) as input					model with 32768 samples (1486 ms) as input				
model	n	layer	filter length & stride	AUC	model	n	layer	filter length & stride	AUC
64 frames	6	1+6+1	256	0.8839	128 frames	7	1+7+1	256	0.8834
128 frames	7	1+7+1	128	0.8899	256 frames	8	1+8+1	128	0.8872
256 frames	8	1+8+1	64	0.8968	512 frames	9	1+9+1	64	0.8980
512 frames	9	1+9+1	32	0.8994	1024 frames	10	1+10+1	32	0.8988
1024 frames	10	1+10+1	16	0.9011	2048 frames	11	1+11+1	16	0.9017
2048 frames	11	1+11+1	8	0.9031	4096 frames	12	1+12+1	8	0.9031
4096 frames	12	1+12+1	4	0.9036	8192 frames	13	1+13+1	4	0.9039
8192 frames	13	1+13+1	2	0.9032	16384 frames	14	1+14+1	2	0.9040
3^n models									
model with 19683 samples (893 ms) as input					model with 59049 samples (2678 ms) as input				
model	n	layer	filter length & stride	AUC	model	n	layer	filter length & stride	AUC
27 frames	3	1+3+1	729	0.8655	81 frames	4	1+4+1	729	0.8655
81 frames	4	1+4+1	243	0.8753	243 frames	5	1+5+1	243	0.8823
243 frames	5	1+5+1	81	0.8961	729 frames	6	1+6+1	81	0.8936
729 frames	6	1+6+1	27	0.9012	2187 frames	7	1+7+1	27	0.9002
2187 frames	7	1+7+1	9	0.9033	6561 frames	8	1+8+1	9	0.9030
6561 frames	8	1+8+1	3	0.9039	19683 frames	9	1+9+1	3	**0.9055**
4^n models									
model with 16384 samples (743 ms) as input					model with 65536 samples (2972 ms) as input				
model	n	layer	filter length & stride	AUC	model	n	layer	filter length & stride	AUC
64 frames	3	1+3+1	256	0.8828	256 frames	4	1+4+1	256	0.8813
256 frames	4	1+4+1	64	0.8968	1024 frames	5	1+5+1	64	0.8950
1024 frames	5	1+5+1	16	0.9010	4096 frames	6	1+6+1	16	0.9001
4096 frames	6	1+6+1	4	0.9021	16384 frames	7	1+7+1	4	0.9026
5^n models									
model with 15625 samples (709 ms) as input					model with 78125 samples (3543 ms) as input				
model	n	layer	filter length & stride	AUC	model	n	layer	filter length & stride	AUC
125 frames	3	1+3+1	125	0.8901	625 frames	4	1+4+1	125	0.8870
625 frames	4	1+4+1	25	0.9005	3125 frames	5	1+5+1	25	0.9004
3125 frames	5	1+5+1	5	0.9024	15625 frames	6	1+6+1	5	0.9041

Table 2. Comparison of various m^n-DCNN models with different input sizes. m refers to the filter length and pooling length of intermediate convolution layer modules and n refers to the number of the modules. Filter length & stride indicates the value of the first convolution layer. In the layer column, the first digit '1' of 1+n+1 is the strided convolution layer, and the last digit '1' is convolution layer which actually works as a fully-connected layer.

4.2 Optimization

We used sigmoid activation for the output layer and binary cross entropy loss as the objective function to optimize. For every convolution layer, we used batch normalization [20] and ReLU activation. We should note that, in our experiments, batch normalization plays a vital role in training the deep models that takes raw waveforms. We applied dropout of 0.5 to the output of the last convolution layer and minimized the objective function using stochastic gradient descent with 0.9 Nesterov momentum. The learning rate was initially set to 0.01 and decreased by a factor of 5 when the validation loss did not decrease more than 3 epochs. A total decrease of 4 times, the learning rate of the last training was 0.000016. Also, we used batch size of 23 for MTAT and 50 for MSD, respectively. In the mel-spectrogram model, we conducted the input normalization simply by dividing the standard deviation after subtracting mean value of entire input data. On the other hand, we did not perform the input normalization for raw waveforms.

5. RESULTS

In this section, we examine the proposed models and compare them to previous state-of-the-art results.

5.1 m^n-DCNN models

Table 2 shows the evaluation results for the m^n-DCNN models on MTAT for different input sizes, number of layers, filter length and stride of the first convolution layer. As

3^n models, 59049 samples as input	n	window (filter length)	hop (stride)	AUC
Frame-level (mel-spectrogram)	4	729	729	0.9000
	5	729	243	0.9005
	5	243	243	0.9047
	6	243	81	**0.9059**
	6	81	81	0.9025
Frame-level (raw waveforms)	4	729	729	0.8655
	5	729	243	0.8742
	5	243	243	0.8823
	6	243	81	0.8906
	6	81	81	0.8936
Sample-level (raw waveforms)	7	27	27	0.9002
	8	9	9	0.9030
	9	3	3	**0.9055**

Table 3. Comparison of three CNN models with different window (filter length) and hop (stride) sizes. n represents the number of intermediate convolution and max-pooling layer modules, thus 3^n times hop (stride) size of each model is equal to the number of input samples.

input type	model	MTAT	MSD
Frame-level (mel-spectrogram)	Persistent CNN [21]	0.9013	-
	2D CNN [14]	0.894	0.851
	CRNN [15]	-	0.862
	Proposed DCNN	**0.9059**	-
Frame-level (raw waveforms)	1D CNN [11]	0.8487	-
Sample-level (raw waveforms)	Proposed DCNN	**0.9055**	**0.8812**

Table 4. Comparison of our works to prior state-of-the-arts

described in Section 3.4, m refers to the filter length and pooling length of intermediate convolution layer modules and n refers to the number of the modules. In Table 2, we can first find that the accuracy is proportional to n for most m. Increasing n given m and input size indicates that the filter length and stride of the first convolution layer become closer to the sample-level (e.g. 2 or 3 size). When the first layer reaches the small granularity, the architecture is seen as a model constructed with the same filter length and sub-sampling length in all convolution layers as depicted in Table 1. The best results were obtained when m was 3 and n was 9. Interestingly, the length of 3 corresponds to the 3-size spatial filters in the VGG net [2]. In addition, we can see that 1-3 seconds of audio as an input length to the network is a reasonable choice in the raw waveform model as in the mel-spectrogram model.

5.2 Mel-spectrogram and raw waveforms

Considering that the output size of the first convolution layer in the raw waveform models is equivalent to the mel-spectrogram size, we further validate the effectiveness of the proposed sample-level architecture by performing experiments presented in Table 3. The models used in the experiments follow the configuration strategy described in Section 3.4. In the mel-spectrogram experiments, 128 mel-bands are used to match up to the number of filters in the first convolution layer of the raw waveform model. FFT size was set to 729 in all comparisons and the magnitude compression is applied with a nonlinear curve, $\log(1 + C|A|)$ where A is the magnitude and C is set to 10.

The results in Table 3 show that the sample-level raw waveform model achieves results comparable to the frame-level mel-spectrogram model. Specifically, we found that using a smaller hop size (81 samples $\approx$ 4 ms) worked better than those of conventional approaches (about 20 ms or so) in the frame-level mel-spectrogram model. However, if the hop size is less than 4 ms, the performance degraded. An interesting finding from the result of the frame-level raw waveform model is that when the filter length is larger than the stride, the accuracy is slightly lower than the models with the same filter length and stride. We interpret that this result is due to the learning ability of the phase variance. As the filter length decreases, the extent of phase variance that the filters should learn is reduced.

5.3 MSD result and the number of filters

We investigate the capacity of our sample-level architecture even further by evaluating the performance on MSD that is ten times larger than MTAT. The result is shown in Table 4. While training the network on MSD, the number of filters in the convolution layers has been shown to affect the performance. According to our preliminary test results, increasing the number of filters from 16 to 512 along the layers was sufficient for MTAT. However, the test on MSD shows that increasing the number of filters in the first convolution layer improves the performance. Therefore, we increased the number of filters in the first convolution layer from 16 to 128.

5.4 Comparison to state-of-the-arts

In Table 4, we show the performance of the proposed architecture to previous state-of-the-arts on MTAT and MSD. They show that our proposed sample-level architecture is highly effective compared to them.

5.5 Visualization of learned filters

The technique of visualizing the filters learned at each layer allows better understanding of representation learning in the hierarchical network. However, many previous works in music domain are limited to visualizing learned filters only on the first convolution layer [11, 12].

The gradient ascent method has been proposed for filter visualization [22] and this technique has provided deeper understanding of what convolutional neural networks learn from images [23, 24]. We applied the technique to our DCNN to observe how each layer hears the raw waveforms. The gradient ascent method is as follows. First, we generate random noise and back-propagate the errors in the network. The loss is set to the target filter activation. Then,

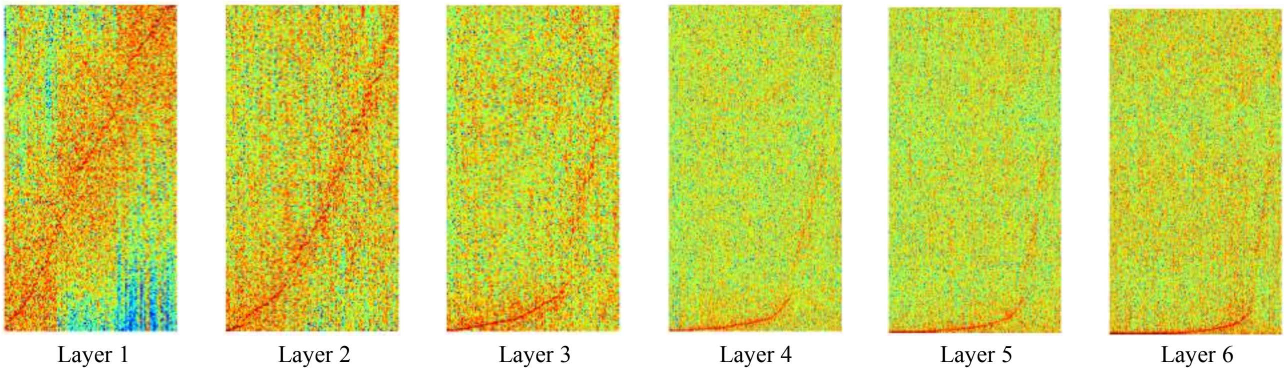


Figure 2. Spectrum of the filters in the sample-level convolution layers which are sorted by the frequency of the peak magnitude. The x-axis represents the index of the filter, and the y-axis represents the frequency. The model used for visualization is 3^9-DCNN with 59049 samples as input. Visualization was performed using the gradient ascent method to obtain the input waveform that maximizes the activation of a filter in the layers. To effectively find the filter characteristics, we set the input waveform estimate to 729 samples which is close to a typical frame size.

we add the bottom gradients to the input with gradient normalization. By repeating this process several times, we can obtain the waveform that maximizes the target filter activation. Examples of learned filters at each layer are in Figure 3. Although we can find the patterns that low-frequency filters are more visible along the layer, estimated filters are still noisy. To show the patterns more clearly, we visualized them as spectrum in the frequency domain and sorted them by the frequency of the peak magnitude.

Note that we set the input waveform estimate to 729 samples in length because, if we initialize and back-propagate to the whole input size of the networks, the estimated filters will have large dimensions such as 59049 samples in computing spectrum. Thus, we used the smaller input samples which can find the filter characteristics more effectively and also are close to a typical frame size in spectrum.

The layer 1 shows the three distinctive filter bands which are possible with the filter length with 3 samples (say, a DFT size of 3). The center frequency of the filter banks increases linearly in low frequency filter banks but it becomes non-linearly steeper in high frequency filter banks. This trend becomes stronger as the layer goes up. This nonlinearity was found in learned filters with a frame-level end-to-end learning [11] and also in perceptual pitch scales such as mel or bark.

6. CONCLUSION AND FUTURE WORK

In this paper, we proposed sample-level DCNN models that take raw waveforms as input. Through our experiments, we showed that deeper models (more than 10 layers) with a very small sample-level filter length and subsampling length are more effective in the music auto-tagging task and the results are comparable to previous state-of-the-art performances on the two datasets. Finally, we visualized hierarchically learned filters. As future work, we will analyze why the deep sample-level architecture works well without input normalization and nonlinear function that compresses the amplitude and also investigate the hierarchically learned filters more thoroughly.

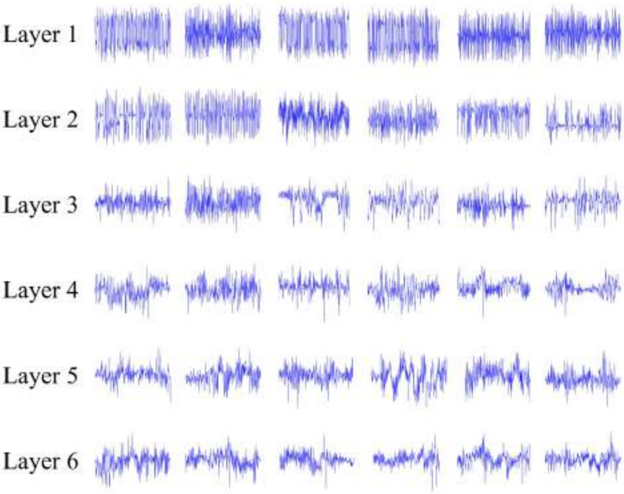


Figure 3. Examples of learned filters at each layer.

Acknowledgments

This work was supported by Korea Advanced Institute of Science and Technology (project no. G04140049) and National Research Foundation of Korea (project no. N01160463).

7. REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[2] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[3] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[4] X. Zhang, J. Zhao, and Y. LeCun, “Character-level convolutional networks for text classification,” in *Advances in neural information processing systems*, 2015, pp. 649–657.

[5] Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush, “Character-aware neural language models,” *arXiv preprint arXiv:1508.06615*, 2015.

[6] D. Palaz, M. M. Doss, and R. Collobert, “Convolutional neural networks-based continuous speech recognition using raw speech signal,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 4295–4299.

[7] D. Palaz, R. Collobert *et al.*, “Analysis of cnn-based speech recognition system using raw speech as input,” Idiap, Tech. Rep., 2015.

[8] R. Collobert, C. Puhrsch, and G. Synnaeve, “Wav2letter: an end-to-end convnet-based speech recognition system,” *arXiv preprint arXiv:1609.03193*, 2016.

[9] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, “Wavenet: A generative model for raw audio,” *CoRR abs/1609.03499*, 2016.

[10] T. N. Sainath, R. J. Weiss, A. W. Senior, K. W. Wilson, and O. Vinyals, “Learning the speech front-end with raw waveform cldnns.” in *INTERSPEECH*, 2015, pp. 1–5.

[11] S. Dieleman and B. Schrauwen, “End-to-end learning for music audio,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 6964–6968.

[12] D. Ardila, C. Resnick, A. Roberts, and D. Eck, “Audio deepdream: Optimizing raw audio with convolutional networks.”

[13] J. Pons, T. Lidy, and X. Serra, “Experimenting with musically motivated convolutional neural networks,” in *IEEE International Workshop on Content-Based Multimedia Indexing (CBMI)*, 2016, pp. 1–6.

[14] K. Choi, G. Fazekas, and M. Sandler, “Automatic tagging using deep convolutional neural networks,” in *Proceedings of the 17th International Conference on Music Information Retrieval (ISMIR)*, 2016, pp. 805–811.

[15] K. Choi, G. Fazekas, M. Sandler, and K. Cho, “Convolutional recurrent neural networks for music classification,” *arXiv preprint arXiv:1609.04243*, 2016.

[16] P. Hamel, S. Lemieux, Y. Bengio, and D. Eck, “Temporal pooling and multiscale learning for automatic annotation and ranking of music audio,” in *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR)*, 2011.

[17] J. Lee and J. Nam, “Multi-level and multi-scale feature aggregation using pre-trained convolutional neural networks for music auto-tagging,” *arXiv preprint arXiv:1703.01793*, 2017.

[18] E. Law, K. West, M. I. Mandel, M. Bay, and J. S. Downie, “Evaluation of algorithms using games: The case of music tagging,” in *ISMIR*, 2009, pp. 387–392.

[19] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, “The million song dataset,” in *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR)*, vol. 2, no. 9, 2011, pp. 591–596.

[20] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” *arXiv preprint arXiv:1502.03167*, 2015.

[21] J.-Y. Liu, S.-K. Jeng, and Y.-H. Yang, “Applying topological persistence in convolutional neural network for music audio signals,” *arXiv preprint arXiv:1608.07373*, 2016.

[22] D. Erhan, Y. Bengio, A. Courville, and P. Vincent, “Visualizing higher-layer features of a deep network,” *University of Montreal*, vol. 1341, p. 3, 2009.

[23] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *European conference on computer vision*. Springer, 2014, pp. 818–833.

[24] A. Nguyen, J. Yosinski, and J. Clune, “Deep neural networks are easily fooled: High confidence predictions for unrecognizable images,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 427–436.

Generating data to train convolutional neural networks for classical music source separation

Marius Miron, Jordi Janer, Emilia Gómez
Music Technology Group, Universitat Pompeu Fabra, Barcelona
firstname.lastname@upf.edu

ABSTRACT

Deep learning approaches have become increasingly popular in estimating time-frequency masks for audio source separation. However, training neural networks usually requires a considerable amount of data. Music data is scarce, particularly for the task of classical music source separation, where we need multi-track recordings with isolated instruments. In this work, we depart from the assumption that all the renditions of a piece are based on the same musical score, and we can generate multiple renditions of the score by synthesizing it with different performance properties, e.g. tempo, dynamics, timbre and local timing variations. We then use this data to train a convolutional neural network (CNN) which can separate with low latency all the renditions of a score or a set of scores. The trained model is tested on real life recordings and is able to effectively separate the corresponding sources. This work follows the principle of research reproducibility, providing related data and code, and can be extended to separate other pieces.

1. INTRODUCTION

Source separation assumes recovering the sources of the signals from a mixture. For audio applications the two main areas are speech enhancement or recognition, and separation of musical sources [1]. Regarding the latter, it allows for a range of interesting applications such as remixing, up-mixing or 3D concerts using VR technologies [2].

In this paper, we study the case of low latency monaural source separation of classical music recordings, where our goal is to extend the instrument enhancement applications developed during the PHENICX project [3, 4] to a low latency scenario, e.g. video streaming applications. In this scenario, we need to provide real-time source separation during a music concert, allowing the listener to focus on specific musical instruments. For that, we can benefit from previous information about the target piece and meta-data associated with it.

In this context, we assume that the sources are harmonic, playing harmonically and rhythmically related phrases, highly correlated in time and frequency which is a challenging scenario if compared to separating between pitched instruments and drums [5]. However, we rely on the assumption that the instruments for a given musical piece are known in advance, and one can include timbre information to guide the separation. This scenario is commonly known as timbre-informed source separation [6].

Matrix decomposition techniques have been traditionally used for timbre-informed music source separation. In the case of Non-negative matrix factorization (NMF), instruments are assigned a set of timbre basis which are previously learned and kept fixed during the separation stage [6]. Although successful, informed NMF approaches are computationally intensive, which makes them difficult to use in a low latency scenario.

Data-driven approaches using deep neural networks involve learning binary or soft masks corresponding to the target sources [7–11]. Moreover, a neural network framework can process short audio windows in streaming in a causal manner which makes these systems suitable for low-latency applications. Furthermore, compared to the NMF, neural networks such as convolutional neural networks or recurrent neural networks have the advantage of modeling the temporal context.

For the source separation approaches based on deep learning, the timbres corresponding to the instruments are learned from multi-microphone tracks. These approaches usually require large amounts of training data, which is scarce for music source separation. In addition, obtaining training data for classical music in form of real-life recordings is difficult because the mixtures should be based on isolated recordings [4, 12, 13]. However, as the classical music repertory has been traditionally assembled with scores, training data can be generated by synthesizing a target score according to several performance factors comprising (and not limited to) tempo, dynamics, timbre of the instruments, and synchronization between musicians, which induce local timing variations. The principle guiding the present study is that these factors differentiate between various renditions and characterize musical performances [14], and training a neural network with such synthetic data generates a more robust model which can be used to separate real-life renditions.

As various classical music pieces are played by different instruments and neural networks architectures are not flexible in accommodating a variable number of sources, it is not possible to train an universal model for classical music. Traditionally, in a timbre-informed case, models

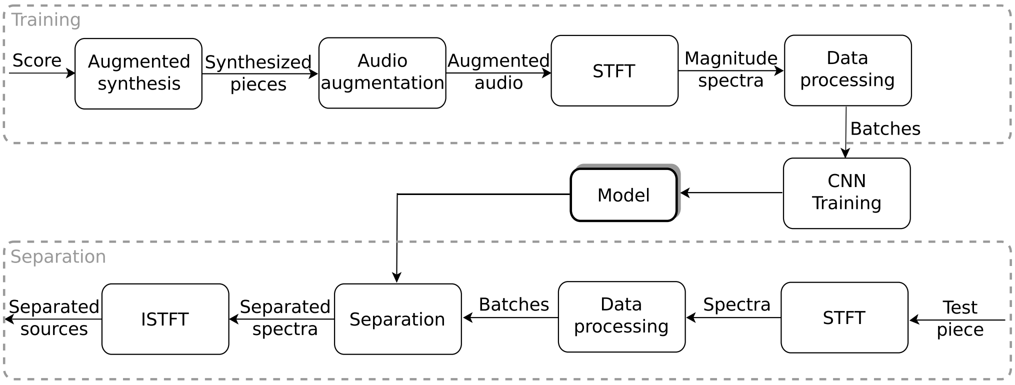


Figure 1. Proposed separation system

are trained for a fixed combinations of instruments [7–11]. For classical music, we constrain the timbre-informed system to the possible combinations of notes and instruments which exist in a set of scores which have the same instruments. We denote this case as score-constrained source separation. This is a more general case than score-informed source separation [15], as score is used solely during the training phase to synthesize renditions.

We propose a timbre-informed and score-constrained system to train neural networks for monaural source separation of classical music mixtures. We generate training data through synthesis of a set of scores by considering the following factors: tempo, timbre, dynamics and circular shifting (local timing variations). In addition, we extend the timbre-informed low latency system we proposed in [11] to the case of score-constrained classical music source separation. As argued in [11], the model we use has considerable less parameters, is easier to train, and can be used in a low latency scenario. We evaluate several training scenarios on Bach chorales pieces played by four instruments.

The remainder of this paper is structured as follows. In Section 2 we discuss the relation of the proposed method with the previous work. We present the architecture of the neural network in Section 3.1, the parameter learning in Section 3.2, the data processing in Section 3.3, and the proposed approaches for generating training data in Section 3.4. The evaluation dataset, method, parameters and results are discussed in Section 4. We conclude with an outlook on the presented system in Section 5.

2. RELATION TO PREVIOUS WORK

As mentioned before, training NMF basis for source separation is carried out either by score synthesis [13, 16] or by analytic approaches which assume learning registers for all considered instruments [6, 15]. In a similar way, we can synthesize the desired pieces or use samples for notes to train a neural network. However, in contrast to NMF, we need to train the network with examples containing the mixtures along with the targeted sources which cover possible cases one might encounter in real-life or in the test dataset. In addition, a deep learning model is optimized and heavily dependent on the training data and needs to cover more examples at the training stage than the parametric approaches embodied by NMF.

As a solution to data scarcity and to improve generalization and robustness, deep learning systems rely on data augmentation techniques by applying transformations to the existing data [17, 18]. Choosing realistic transformations depends on the possible axes on which data varies and on the task, e.g. pitch shifting, time stretching, loudness, randomly mixing different recordings or background noise. Although our approach can be seen as a data augmentation strategy, we generate new data according to transformations that make more sense for source separation, instead of augmenting existing data with techniques popular in other classification tasks [17, 18]. Furthermore, except circular shifting [7], we use different transformations than previous deep learning approaches. We consider music samples played with different dynamics and by different instruments (timbre), which is different than simply changing the loudness or the amplitude of a sample. Additionally, we mix the audio files and not the resulting spectrograms to account for phase cancellation.

Besides [10], deep learning approaches are evaluated in a cross-validation manner [7, 9, 11]. However, we claim that due to the small size of the classical music datasets used for evaluation, these methods are more likely to perform poorly in real-life scenarios where we can have different tempos, timbre, or dynamics. Therefore, varying training data with these factors adds robustness.

3. METHOD

The diagram for the proposed system is depicted in Figure 1. For the training stage, we depart from the score of a given piece which we synthesize at various combinations of tempos, timbres and dynamics. We then generate additional data by mixing circular shifted versions of the audio tracks for the corresponding sources. More details on data generation are given in Section 3.4.

3.1 Network architecture

The convolutional neural network (CNN) used in this paper is a variation of the one we formulated in [11] and comprises two convolutional layers, a dense ”bottleneck” layer having a low number of units, a second dense layer followed by the inverse of the first two layers for each of the target sources. We kept similar filter shapes in order to preserve a low number of parameters and the low latency of the model.

The input of the network is a STFT magnitude spectrogram $\mathbf{X} \in \mathbb{R}^{TF}$, where T is the number of time frames (the time context modeled) and F is the number of frequency bins. This input is passed through a vertical convolution layer comprising $P_1 = 30$ filters of size $(1, 30)$ with stride 5, thus yielding the output $\mathbf{X_1} \in \mathbb{R}^{P_1 T F_1}$, where $F_1 = (F - 30)/5 + 1$. Then, we have a horizontal convolution comprising $P_2 = 30$ filters of size $(\frac{2}{3} \cdot T, 1)$ with stride 1 which yields the output $\mathbf{X_2} \in \mathbb{R}^{P_2 T_2 F_1}$, where $T_2 = (T - \frac{2}{3} \cdot T)/1 + 1$. This layer is followed by a dense "bottleneck" layer of size $F_3 = 256$, which has an input of size $P_2 \cdot T_2 \cdot F_1$.

As we need to reconstruct the output for each of the sources $j = 1 : J$, where J is the total number of separated sources, we perform the inverse operations for the horizontal and the vertical convolutions. To match the dimensions needed to compute the inverse of the second layer, we need to create a dense layer of size $P_2 \cdot T_2 \cdot F_1$ features for each source, thus having $J \cdot P_2 \cdot T_2 \cdot F_1$ units. Consequentially, for each of the estimated sources we perform the inverse operation of the convolution layers, i.e. the deconvolution, using the same parameters learned by these layers. The final inverse layer yields a set of estimations $\mathbf{E_j} \in \mathbb{R}^{TF}$, with $j = 1 : J$.

The convolutional layers have a linear activation function and the dense ones a rectified linear unit activation function, as in [11].

3.2 Parameter learning

Similarly to [7,11], the network yields the estimations $\mathbf{E_j} \in \mathbb{R}^{TF}$ from which we derive the soft masks $\mathbf{M_j} \in \mathbb{R}^{TF}$, for each source $j = 1 : J$:

$$\mathbf{M_j} = \frac{|\mathbf{E_j}|}{\sum_{j=1}^{J} |\mathbf{E_j}|} \tag{1}$$

Then, the soft masks are multiplied with the original magnitude spectrogram to obtain the corresponding estimated magnitude spectrograms for the sources: $\hat{\mathbf{X}}_\mathbf{j} = \mathbf{M_j} \times \mathbf{X}$.

The parameters of the network are updated using minibatch Stochastic Gradient Descent with *AdaDelta* algorithm [19] according to the following loss function which minimizes the Euclidean distance between the estimated $\hat{\mathbf{X}}_\mathbf{j}$ and target $\mathbf{X_j}$ sources: $Loss = \sum_{j=1}^{J} ||\hat{\mathbf{X}}_\mathbf{j} - \mathbf{X_j}||^2$

Because the target sources are harmonic, highly correlated and playing consonant musical phrases, we do not include the dissimilarity cost between the sources as in [11].

3.3 Data processing

a) Training: Generating training data assumes slicing the spectrograms of the mixture $\mathbf{X}(t, f)$ and the target spectrograms $\mathbf{X}_j(t, f)$ in overlapping blocks of size T, for the time frames $t = 1 : \hat{T}$ and frequency bins $f = 1 : F$, where $\hat{T}$ is the total number of time frames and F is the number of frequency bins of $\mathbf{X}$. We summarize the procedure in Algorithm 1.

Algorithm 1 Generating training data

```
for each piece in the training set do
    Compute STFT of the piece
    Initialize total number of blocks to $B = \frac{\hat{T}-T}{O}$, where $O$ is the step size.
    for b=1:B do
        Slice $\mathbf{X}^b$ = $\mathbf{X}(b * O : b * O + T, :)$
        Slice $\mathbf{X}_j^b$ = $\mathbf{X}_j(b * O : b * O + T, :)$
    end for
end for
Generate batches by randomly grouping blocks.
```

b) Separation: Separating a target signal involves slicing a magnitude spectrogram. Then, we obtain an estimation for the sources by feed forwarding the blocks through the network, and we overlap-add the blocks to obtain the magnitude spectrograms. The steps are presented in the Algorithm 2.

Algorithm 2 Separation of a piece

```
Compute complex valued STFT; keep the phase.
Initialize total number of blocks to $B = \frac{\hat{T}-T}{O}$, where $O$ is the step size.
for b=1:B do
    Slice $\mathbf{X}^b$ = $\mathbf{X}(b * O : b * O + T, :)$
    Feed-forward $X^b$ through the CNN, obtaining the magnitude spectrograms $\hat{X}_j^b$, for the sources $j = 1 : J$.
end for
for j=1:J do
    Compute $\hat{X}_j$ by overlap-adding $\hat{X}_j^b$.
end for
Use the phase of the original signal and compute the estimated sources with the inverse overlap-add STFT.
```

If we consider that the sources $y_j(t), j = 1 : J$ are linearly mixed, such that $x(t) = \sum_{j=1}^{J} y_j(t)$, then we can use the original phase of the audio mixture to obtain the signals associated to the sources $y_j(t)$, with an inverse overlap-add STFT, as in [13].

3.4 Data generation

The proposed system is trained with data generated by synthesizing a set of scores at different tempos, using samples of different timbres and dynamics, and then applying circular shifting to the resulting audio files, to account for local timing variations. In this paper, we consider four factors:

a) Tempo: The renditions of a classical music piece vary in terms of tempo. Therefore, in order to make the model more robust to variations in tempo, we synthesize the score at different tempos, i.e. we adjust the note duration for each note corresponding to each instrument in the score. The possible tempo variations are represented in the vector $a = \{a_1, a_2, \ldots , a_A\}$, where A is the total number of considered tempos.

b) Timbre: Different recording conditions, instruments, players or playing styles can yield differences in timbre. A single timbre variation comprises samples of the notes in the designated range of an instrument played by a different musician and recorded in different conditions. All timbre variations are stored in the vector $c = \{c_1, c_2, \dots, c_C\}$, where C is the total number of timbre variations. Hence, when synthesizing a note of a given instrument we can choose between the c samples corresponding to a different timbre.

c) Dynamics: The dynamics of a piece can change between renditions of a piece. Furthermore, variations in dynamics induce changes in loudness and timbre. Thus, we synthesize using samples representing various levels of dynamics in order to make our model more robust to this variable. The dynamics variations are represented in the vector $d = \{d_1, d_2, \dots, d_D\}$, where D is the dynamic range.

d) Circular shifting: In contrast to tempo changes which account for global tempo variations, circular shifting accounts for local timing variations. The synthetic pieces lack human expressiveness or even small errors which one might encounter in a real performance and for which we try to account for using this transformation. Circular shifting takes place after the synthesis of the audio and is applied to each of the target sources. Considering an audio signal of S samples, a circular shift is a permutation σ with $\hat{S}$ samples such that $\sigma(i) \equiv (i + \hat{S}) \mod S$, for all samples $i = 1 : S$.

We can have various combinations between different shifting steps for the sources. For each combination we generate a new mixture by summing up the circular shifted audio tracks. The possible circular shifts are represented in the vector $e = \{e_1, e_2, \dots, e_E\}$, where E is the total number of considered circular shifts.

The space comprising all possible combinations between the considered variables and $j = 1 : J$ target instruments is the Cartesian product $a \times c \times d \times e \times j$, having in total $A{\cdot}C{\cdot}D{\cdot}E{\cdot}J$ possibilities. If the number of combinations is too large and we can not generate all of them, we randomly sample from the Cartesian space.

In the Algorithm 3 we detail the procedure used to generate a rendition, i.e. a multi-track recording comprising audio vectors x_j for each instrument $j = 1 : J$. We depart from a given score which has a set of notes $n_j = 1 : N_j$, where N_j is the total number of notes for instrument j. Then we synthesize each note considering a given combination comprising a tempo $\hat{a}$, a set timbres $\hat{c} = (\hat{c}_1, \dots, \hat{c}_j)$, dynamics $\hat{d} = (\hat{d}_1, \dots, \hat{d}_j)$ and circular shifting $\hat{e} = (\hat{e}_1, \dots, \hat{e}_j)$ for each instrument $j = 1 : J$.

Algorithm 3 Generating a rendition from a score

```
Initialize the tempo $\hat{a}$, timbre $\hat{c}$, dynamics $\hat{d}$ and circular
shifting $\hat{e}$ variables.
for $j = 1 : J$ do
    Initialize the audio vector $x_j$
    for each note $n_j = 1 : N_j$ do
        Adjust the note duration in the score to the tempo $\hat{a}$.
        Query the database/audio engine for a sample of timbre
        $\hat{c}_j$ with the dynamics $\hat{d}_j$.
        Synthesize the audio vector corresponding to the given
        note $n_j$ and paste it in $x_j$ at the onset and offset times.
    end for
    Apply the circular shift $\hat{e}_j$ to $x_j$.
end for
```

4. EVALUATION

4.1 Datasets

For evaluation purposes we use ten Bach chorales from the Bach10 dataset, played by bassoon, clarinet, saxophone, violin, in the Bach10 dataset [12] which has been widely used in the evaluation of source separation. Each piece has a duration of ≈ 30 seconds and is accompanied by the original score and a version of this score which is aligned with the rendition.

We synthesize the scores in the Bach10 dataset with two methods:

a) Sibelius: We use the library provided by software Sibelius [1] , which uses sample-based synthesis. In this case we have $C = 1$ timbre and $D = 1$ for dynamics. Moreover, we use three levels of tempo $a = \{80, 100, 120\}$ BPM, and three of circular shift $e = \{0, 0.1, 0.2\}$ seconds. The dataset is made available through Zenodo [2] .

b) RWC: In this experiments we query the RWC database [20] for samples corresponding to the notes played by the instruments in Bach10 for the original score $a = \{100\}$ BPM. The samples are played by three different musicians $c = \{1, 2, 3\}$, at three levels of dynamics $d = \{$*forte, mezzo, piano*$\}$, and various styles. For this experiment we picked the *normal* style of playing. The circular shifting considered for this method is $e = \{0, 0.1, 0.2\}$ seconds.

Because we want to isolate the influence of timbre and dynamics from the influence of tempo, we also synthesize the ten pieces using the score perfectly aligned with the audio using the synthesis methods **a** and **b**,

4.2 Evaluation setup

a) Evaluation metrics: We used the evaluation framework and the metrics are described in [21] and [22] : *Source to Distortion Ratio* (SDR), *Source to Interference Ratio* (SIR), and *Source to Artifacts Ratio* (SAR).

b) Parameter tuning: For the STFT we used a Blackman-Harris window. We experimentally set the length of the window to 4096 samples, which, at a sampling rate of 44.1 KHz corresponds to 96 milliseconds (ms), and a hop size of 512 samples ($11ms$). We observed that, for the Bach10 dataset, the higher the STFT resolution the better the results in separation, especially for the instruments which have a lower range, such as bassoon.

We kept the time context modeled by the CNN to $T = 30$ frames, which showed best performance in [11] with the step size $O = 25$. The number epochs, which represents the number of times all the examples of the dataset are seen by the network, is experimentally determined for each training experiment. As a general rule, we stop training if the cost between two epochs drops below 0.05. The size of a mini-batch is set to 32.

Additionally, various hypotheses were tested and were not proven to improve the separation: CNN architectures having separate filters for each instrument, l_2 regularization and dropout for the dense layers, and Kullback-Leibler

[1] `http://www.avid.es/sibelius`

[2] `https://zenodo.org/record/321361#.WKxZKd-i7J8`

and Itakura Saito distances instead of Euclidean distance in the cost function.

c) Hardware and software tools: Our work follows the principle of research reproducibility, so that the code used in this paper is made available [3] . It is built on top of Lasagne, a framework for neural networks using Theano [4] . We ran the experiments on a computer with GeForce GTX TITAN X GPU, Intel Core i7-5820K 3.3GHz 6-Core Processor, X99 gaming 5 x99 ATX DDR44 motherboard.

4.3 Experiments

4.3.1 Convolutional Neural Networks

In order to test various data generation approaches, we train the CNN system in Section 3.1 with different data:

a) CNN Bach10: We train with all the 10 pieces in the Bach10 dataset.

b) CNN leave one out (LOOCV) on Bach10: We train with nine pieces of Bach10 and test on the remaining piece, repeating this for all the 10 pieces of the dataset.

c) CNN Sibelius: We train with all the synthetic pieces in the generated dataset described in Section 4.1a.

d) CNN Sibelius GT: We train with all the synthetic pieces in generated dataset described in Section 4.1a, synthesized with the score perfectly aligned with the rendition.

e) CNN RWC: We train with all the synthetic pieces in the generated dataset described in Section 4.1b. Since the number of the possible combinations between the factors a, c, d, e is too large, we randomly sample 400 points.

f) CNN RWC GT: We train with all the synthetic pieces in the generated dataset described in Section 4.1b, synthesized with the score perfectly aligned with the rendition. In this case, because there are less factors to vary, we randomly sample 50 points.

4.3.2 Score-constrained NMF

We compare the proposed approaches with an NMF timbre-informed system based on the multi-source filter model [4, 6] which includes timbre models trained on the RWC dataset. Because we deal with a score-constrained scenario and for a fair comparison, the gains of the NMF are restricted to the notes in the score, without taking into account the time when the notes are played. Thus, each row of the gains matrix corresponding to a note is set to 1 if a given instrument plays a note from the score. The other values in the gains matrix are set to 0 and do not change during computation, while the values set to 1 evolve according to the energy distributed between the instruments. The NMF parameters are kept fixed as in [4]: 50 iterations for the NMF, beta-divergence distortion $\beta = 1.3$, STFT length of window of $96ms$, and a hop size of $11ms$.

4.4 Results and discussion

We present the results of the evaluated approaches in Section 4.3.1 and Section 4.3.2 on the Bach10 and Sibelius datasets. The separated audio files and the computed measures for each file and instrument are made available through Zenodo [5] .

We verify that when the training and test datasets coincide, the CNN approach achieves the best performance. We denote this case as ”Oracle”.

4.4.1 Separation results with real recordings: Bach10

We evaluate the considered approaches on the Bach10 dataset. Overall results are presented in Figure 2, while instrument specific ones are provided in Figure 3.

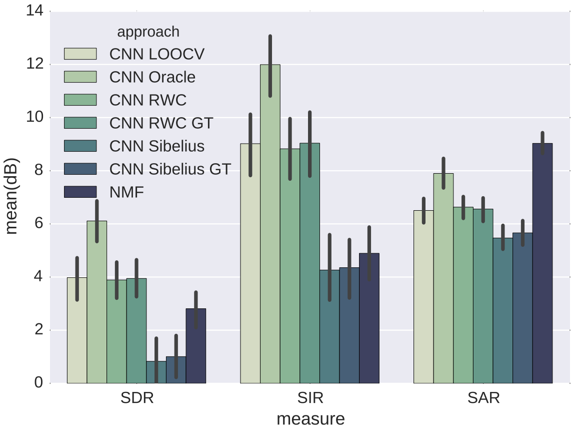


Figure 2. Results in terms of SDR, SIR, SAR for Bach10 dataset and the considered approaches: CNN and NMF [6]

The LOOCV approach is trained with examples from the same dataset, having the same timbre and style, thus achieves $\approx 4dB$ in SDR. This illustrates the fact that training the neural network on similar pieces, played by the same musicians in the same recording conditions is beneficial for the system.

The approach CNN RWC, which involves synthesizing the original score with samples comprising a variety of instrument timbres and dynamics, yields as good results as the LOOCV approach $\approx 4dB$ SDR. On the other hand, if the training set comprises less samples and less variations in timbre and dynamics, the we have considerable lower results, as in the CNN Sibelius approach which has $\approx 1dB$ SDR. In fact, the CNN RWC approach has higher SIR than CNN Sibelius ($9dB$ compared with $4dB$). Thus, learning to separate more diverse examples reduces the interference.

Synthesizing the score perfectly aligned with the rendition does not improve the results for the considered approaches: CNN RWC GT and CNN Sibelius GT. For this particular CNN architecture and the modeled time context ($300ms$), synthesizing the original score with circular shifting to compensate for local timing variation achieves results as good as the ideal case when synthesized a perfectly aligned score. This is encouraging considering that a perfectly aligned score is difficult to obtain. Furthermore, a score-following system introduces additional latency.

[3] `https://github.com/MTG/DeepConvSep`

[4] `http://lasagne.readthedocs.io/en/latest/`Lasagne and `http://deeplearning.net/software/theano/`Theano

[5] `http://zenodo.org/record/344499#.WLbagSMrIy4`

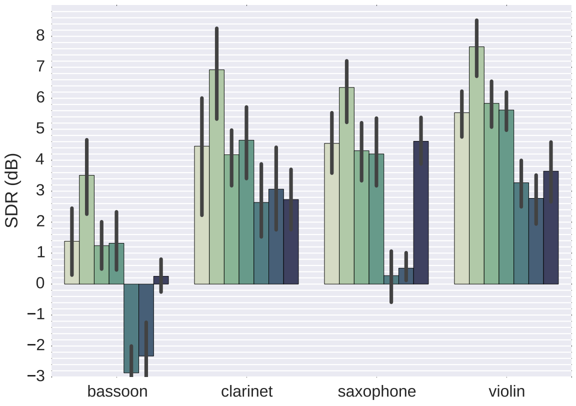


Figure 3. Results for each instrument in terms of SDR for Bach10 dataset and the considered approaches: CNN and NMF [6]

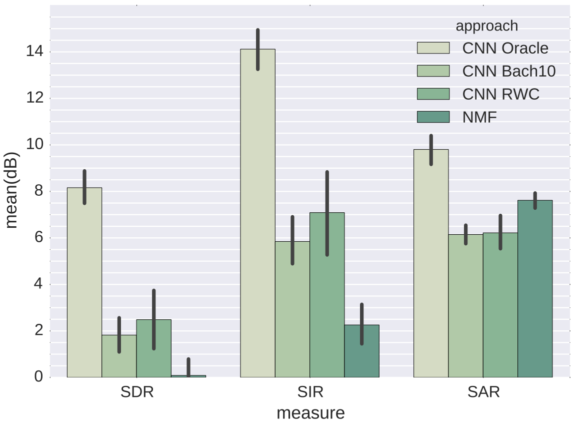


Figure 4. Results in terms of SDR, SIR, SAR for Sibelius dataset and the considered approaches: CNN and NMF [6]

The score-constrained NMF separation which uses timbre models trained on the RWC dataset, has lower SDR than proposed approach CNN RWC. The NMF system has higher SAR, less artifacts, at the expense of having lower SIR, hence more interference between the instruments.

As seen in Figure 3, the separation for all instruments benefits from having examples with more diverse timbre or dynamics in the dataset, as CNN RWC has higher SDR than the CNN Sibelius across all instruments. The results for bassoon are lower across all approaches. This is in line with the results yielded by other state of the art approaches evaluated on this dataset [6,12] and can be due to the lower register of this instrument and the poor resolution of STFT for lower frequencies.

4.4.2 *Separation results with synthesized recordings: Sibelius*

We now evaluate the considered approaches on the synthesized Sibelius dataset. Because ”GT” approaches do not differ from their baselines, we decide not to include them here. The overall results are presented in Figure 4.

When tested on Sibelius dataset which has different tempos, timbres, dynamics than the training Bach10 dataset, the approach CNN Bach10 decreases in SDR with $\approx 4dB$. In fact, all data driven approaches have lower performance on unseen data. This raises questions on the validity of cross-fold evaluation methodology of source separation using small datasets.

As seen in Figure 4, the CNN RWC and NMF yield lower results on Sibelius synthetic dataset than on the real-life renditions of Bach10. In this case, the CNN RWC is $2.5dB$ higher in SDR than the score-constrained NMF. This is in line with the results obtained in Section 4.4.1. Thus, training this CNN with synthetic data of different timbre, dynamics and circular shifting, has better performance than NMF method, being less computationally intensive.

The results for the Oracle in Section 4.4.2 (Figure 2) are higher than Section 4.4.1 (Figure 4), because the synthesized dataset Sibelius lacks the diversity of dynamics and local tempo variations present in real-life performances of Bach10, which are more difficult to model.

We assess the computation time of the proposed framework, implemented in Python, in comparison with the NMF framework [4, 6], implemented in Matlab, on a 2013 MacBook Pro with 2.5Gz Intel Core I5 and 16Gb RAM. Separating with CNN took on average 0.76 of the length of the audio, while the NMF framework took 4.6 of the length of the audio.

5. OUTLOOK

We proposed a method to generate training data for timbre-informed source separation methods using neural networks, in the context of classical music. We departed from a set of scores and we synthesized new renditions by varying tempo, timbre, dynamics and local time variations.

We tested two synthesis approaches on real-life performances and on synthesized pieces of Bach chorales. We showed that evaluating in a cross-validation manner on a small dataset yields results that can not be generalized on different renditions which depart from the same score.

We underline the importance of timbre and dynamics in generating training data. Correspondingly, the approach having more varied timbre and dynamics achieved higher performance, surpassing a score-constrained NMF method [4, 6]. The proposed approach is similar to data augmentation and makes the model more robust to variations which one can expect in real cases. Thus, having a more diverse training set avoids overfitting.

As future work, we have to determine the optimum number of random samples to be used for training, as a trade-off between performance and training time. Then, we plan on testing the current framework on more complex mixtures, as orchestral music. Furthermore, the current method can be extended to a score-informed scenario by having the score as input to the network.

Acknowledgments

The TITANX used for this research was donated by the

NVIDIA Corporation. This work is partially supported by the Spanish Ministry of Economy and Competitiveness under CASAS project (TIN2015-70816-R). We thank Agustin Martorell for his help with Sibelius and Pritish Chandna for his useful feedback.

6. REFERENCES

[1] E. Vincent, C. Févotte, R. Gribonval, L. Benaroya, X. Rodet, A. Röbel, E. Le Carpentier, and F. Bimbot, "A tentative typology of audio source separation tasks," in *4th Int. Symp. on Independent Component Analysis and Blind Signal Separation (ICA)*, 2003, pp. 715–720.

[2] J. Janer, E. Gómez, A. Martorell, M. Miron, and B. de Wit, "Immersive orchestras: audio processing for orchestral music vr content," in *Games and Virtual Worlds for Serious Applications (VS-Games), 2016 8th International Conference on*. IEEE, 2016, pp. 1–2.

[3] E. Gómez, M. Grachten, A. Hanjalic, J. Janer, S. Jordà, C. F. Julià, C. C. S. Liem, A. Martorell, M. Schedl, and G. Widmer, "Phenicx: Performances as highly enriched and interactive concert experiences," in *SMC*, 2013.

[4] M. Miron, J. Carabias-Orti, J. Bosch, E. Gómez, and J. Janer, "Score-informed source separation for multichannel orchestral recordings," *Journal of Electrical and Computer Engineering*, vol. 2016, 2016.

[5] T. Virtanen, "Monaural sound source separation by nonnegative matrix factorization with temporal continuity and sparseness criteria," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 3, pp. 1066–1074, 2007.

[6] J. J. Carabias-Orti, T. Virtanen, P. Vera-Candeas, N. Ruiz-Reyes, and F. J. Canadas-Quesada, "Musical Instrument Sound Multi-Excitation Model for Non-Negative Spectrogram Factorization," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 6, pp. 1144–1158, Oct. 2011.

[7] P.-S. Huang, S. D. Chen, P. Smaragdis, and M. Hasegawa-Johnson, "Singing-voice separation from monaural recordings using robust principal component analysis," in *ICASSP*. IEEE, mar 2012, pp. 57–60.

[8] E. Grais, M. Sen, and H. Erdogan, "Deep neural networks for single channel source separation," in *ICASSP*. IEEE, may 2014, pp. 3734–3738.

[9] A. Simpson, G. Roma, and M. Plumbley, "Deep karaoke: Extracting vocals from musical mixtures using a convolutional deep neural network," in *International Conference on Latent Variable Analysis and Signal Separation*. Springer, 2015, pp. 429–436.

[10] S. Uhlich, F. Giron, and Y. Mitsufuji, "Deep neural network based instrument extraction from music," in *ICASSP*. IEEE, 2015, pp. 2135–2139.

[11] P. Chandna, M. Miron, J. Janer, and E. Gómez, "Monoaural audio source separation using deep convolutional neural networks," *International Conference on Latent Variable Analysis and Signal Separation*, 2017.

[12] Z. Duan and B. Pardo, "Soundprism: An online system for score-informed source separation of music audio," *IEEE Journal of Selected Topics in Signal Processing*, pp. 1–12, 2011. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5887382

[13] J. Fritsch and M. Plumbley, "Score informed audio source separation using constrained non-negative matrix factorization and score synthesis," *ICASSP*, pp. 888–891, 2013.

[14] G. Widmer and W. Goebl, "Computational models of expressive music performance: The state of the art," *Journal of New Music Research*, vol. 33, no. 3, pp. 203–216, 2004.

[15] S. Ewert and M. Müller, "Score-Informed Voice Separation For Piano Recordings." *12th International Society for Music Information Retrieval Conference*, no. 12, pp. 245–250, 2011.

[16] J. Ganseman, P. Scheunders, G. Mysore, and J. Abel, "Source separation by score synthesis." in *International Computer Music Conference*, 2010.

[17] J. Schlüter and T. Grill, "Exploring data augmentation for improved singing voice detection with neural networks," in *16th International Society for Music Information Retrieval Conference*, 2015.

[18] J. Salamon and J. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 279–283, 2017.

[19] M. D. Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[20] M. Goto, "Development of the RWC music database," in *ICA*, 2004, pp. 553–556.

[21] E. Vincent, R. Gribonval, and C. Fevotte, "Performance measurement in blind audio source separation," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, no. 4, pp. 1462–1469, jul 2006.

[22] V. Emiya, E. Vincent, N. Harlander, and V. Hohmann, "Subjective and Objective Quality Assessment of Audio Source Separation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 7, pp. 2046–2057, Sep. 2011.

EVALUATION OF AN ACOUSTIC INTERFACE FOR TREMOR ANALYSIS

Marian Weger, David Pirrò, Robert Höldrich
Institute for Electronic Music and Acoustics (IEM)
University of Music and Performing Arts, Graz, Austria
{weger,pirro,hoeldrich}@iem.at

ABSTRACT

We present the evaluation of a sonification approach for the acoustic analysis of tremor diseases. The previously developed interactive tool offers two methods for sonification of measured 3-axes acceleration data of patients' hands. Both sonifications involve a bank of oscillators whose amplitudes and frequencies are controlled by either frequency analysis similar to a vocoder or Empirical Mode Decomposition (EMD) analysis. In order to enhance the distinct rhythmic qualities of tremor signals, additional amplitude modulation based on measures of instantaneous energy is applied. The sonifications were evaluated in two experiments based on pre-recorded data of patients suffering from different tremor diseases. In Experiment 1, we tested the ability to identify a patient's disease by using the interactive sonification tool. In Experiment 2, we examined the perceptual difference between acoustic representations of different tremor diseases. Results indicate that both sonifications provide relevant information on tremor data and may complement already available diagnostic tools.

1. INTRODUCTION

Tremor is a movement disorder which produces involuntary rhythmic oscillation movements of a body part [1]. As it can be caused by various neurological diseases [2], a correct diagnosis is quickly needed to choose the right therapy. Each of these diseases evokes a specific movement pattern which can be recognized visually by specialized neurologists. This visual diagnosis, however, is unreliable and common approaches for additional ex-post analysis of videos or measured sensor data are time-consuming and can not be easily integrated into daily clinical practice. A sonification has the advantage that it provides an auditory representation which is continuously following the spectral characteristics of the tremor and thus allows to keep track of the time-dependent spectral structures. Real-time sonification of tremor movement data could therefore become a promising extension to already available diagnostic tools. Sonification has recently been successfully used for for therapy of Parkinsonian tremor [3,4].

In a follow-up to our previous research [5], we developed two sonification methods for tremor analysis which are described in detail in [6]. These are intended to be used interchangeably dependent on tremor characteristics and personal preference. Both sonifications and the corresponding graphical user interface are implemented in Pure Data[1]. The interface is targeted towards real-time use as a supplementary medical tool in order to improve diagnostic quality. It is employed to extract relevant features of the tremor signal, which are exposed aurally by the developed sonification algorithms. The resulting feature space is high dimensional and therefore predestined for aural rendering in preference to visual representations. This tool, however, aims not at providing definite answers nor a final diagnosis of the disease.

In a previous pilot study (see [6]), test participants were asked to identify the tremor diseases of patients by using the interactive sonification interface on pre-recorded movement data of 30 patients who divided equally into three different groups of diseases (Parkinsonian, Essential, and Psychogenic tremor). Participants used headphones and obtained prior training with the same set of patients. On average, participants reached 61 % correct diagnoses, which is far above chance (1/3). According to participating neurologists, the proposed interface facilitates an insight in the movement pattern of an examined tremor without visual tools. An interactive switch between sonifications did not improve overall sensitivity. However, test participants positively welcomed the possibility to parametrize the sonifications to personal preference.

In retrospective, this pilot study suffered from unclear data and immature experimental design: It was based on a small sample size, clinical reference diagnoses were not perfectly reliable, and test participants were trained with the same set of patients' tremor data as used in the experiment. The results are therefore of limited significance. As a consequence, we carried out an extended study which is presented here. It includes more test participants and a larger dataset of patients with confirmed diagnosis.

This article is structured as follows. After this introduction, Sec. 2 gives an overview of the technical setup for data acquisition and the two different sonification methods. One used frequency analysis (Vocoder sonification, Sec. 2.1), while the other is based on Empirical Mode Decomposition analysis (EMD sonification, Sec. 2.2). These sonifications were evaluated with real patients' data in an identification test with the interactive audiovisual user interface as well as in a triangle discrimination test (Sec. 3).

[1] Pure Data (Pd): `http://puredata.info/`

Finally, in Sec. 4, we summarize our findings and give an outlook on future work.

Accompanying sound examples can be found on the project web page [7]. These include stereo recordings of both sonifications with two patients of each tremor type.

2. SONIFICATION APPROACH

Both sonification methods share the basic technical setup as well as some fundamental data conditioning steps.

Movement data is recorded by 3-axis accelerometers [2] attached to the patient's hands and sampled at 1 kHz. Signals are recorded with CED Spike2 [3] and pre-processed in Matlab. A DC removal and 70 Hz low pass filter is applied on the acceleration signal in order to cover the typical frequency range of pathological tremor (predominantly $3-15$ Hz). Both left and right arm sensors are individually sonified.

As strong amplitude variations can occur between different measurements, Automatic Gain Control (AGC) is applied to the input signal at different stages in both sonifications. The measurement data is further conditioned by a Principal Component Analysis (PCA) [8]. In the context of the presented sonifications, the first principal component is projected on the multichannel data to retrieve the monophonic input signal $x[t]$ (see [5] for a more detailed description).

We will briefly describe the basic sonification algorithms. For further information, please consult [6].

2.1 Vocoder Sonification

The first sonification method is similar to a vocoder.

By using a sliding window FFT, the input signal is divided into 5 frequency bands ($2-4$ Hz, $4-6$ Hz, $6-9$ Hz, $9-13$ Hz, and $13-20$ Hz). Center frequencies and bandwidths have been selected based on experience with the spectra of different tremor types.

Then, the normalized energies in the individual bands are used to control the amplitudes of 5 sinusoidal oscillators which are tuned harmonically to each other, i.e., following the harmonic series (f_0 , $2 \cdot f_0$, etc.). A Frequency Modulation (FM) with the smoothed half-wave rectified input signal $x[t]$ can be applied optionally. This results in a time-varying fundamental frequency of $f_i(t)$ instead of a constant f_0.

The sum of the five oscillator signals is finally amplitude modulated by the variably smoothed half-wave rectified input signal $x[t]$.

The Vocoder sonification produces a harmonic complex, evoking a clear, optionally time-varying pitch percept (compare sound examples [7]). The time-varying timbral character resembles vocal formants whereas the overall amplitude modulation adds a rhythmic dimension.

2.2 EMD Sonification

The second sonification is based on Empirical Mode Decomposition analysis.

EMD was originally developed by [9] to analyze non-stationary and non-linear signals. The idea of EMD is that complex data sets can be decomposed into a finite (and often small) number of so-called Intrinsic Mode Functions (IMFs). Each IMF represents one mode of the signal. The higher the index of an IMF, the lower its frequency components.

In contrast to Fourier analysis where a signal is decomposed into a set of pre-defined base functions, the EMD obtains the base functions adaptively from the signal. A perfect reconstruction of the original signal is possible via summation of the contained IMFs and the resulting residual signal. The basic EMD algorithm is explained in [9–12].

For each individual IMF, the instantaneous phase, frequency, and amplitude can be obtained from the Hilbert transform. In conjunction with the EMD, this is called the Hilbert-Huang Transform (HHT) [9, 13]. The HHT has been proposed for tremor analysis in recent studies, e.g., [14–16].

For the sonification, only the first five IMFs of the input signal $x[t]$ are determined via EMD. Eventually, these IMFs are individually leveled by AGC.

Each IMF then controls the frequency and amplitude of an individual sinusoidal oscillator. Although both the instantaneous amplitude and the frequency can be computed at any time by using the Hilbert transform, we used the generalized zero-crossing method [17] for the frequency, as it was found to provide more stable results. The determined frequencies are then multiplied by a user-controlled constant factor to map the low tremor frequencies to the audible range. Each oscillator is then individually amplitude modulated by the smoothed half-wave rectified IMF signal itself, in order to display the original tremor frequency range as a superposition of rhythmic structures.

Finally, the output signal of the sonification is formed by the sum of these four signals.

Due to the specific time-varying characteristics of the tremor signals, the sonic result of the EMD-based sonification resembles the sound of singing birds (compare sound examples [7]). The register of each "bird" is dependent on the frequency and amplitude of the corresponding IMF.

3. EVALUATION STUDY

The presented sonification methods as well as the interactive interface were evaluated on the basis of pre-recorded movement data of 97 patients with confirmed diagnosis. The clinical tremor data were collected by the Medical University of Graz and UCL Institute of Neurology London between 2012 and 2013. Hand acceleration data were recorded for both hands simultaneously in rest (arms hanging) and posture (arms outstretched) condition. The patients divide unevenly into four tremor types: 24× Parkinsonian, 19× Essential, 36× Psychogenic, and 18× Dystonic tremor. To force an even distribution, we randomly se-

[2] Biometrics ACL300 (mass: 10 g, accuracy: ±2 % FS): `http://www.biometricsltd.com/accelerometer.htm` connected to CED 1401 interface.

[3] CED Spike2: `http://ced.co.uk/products/spkovin`

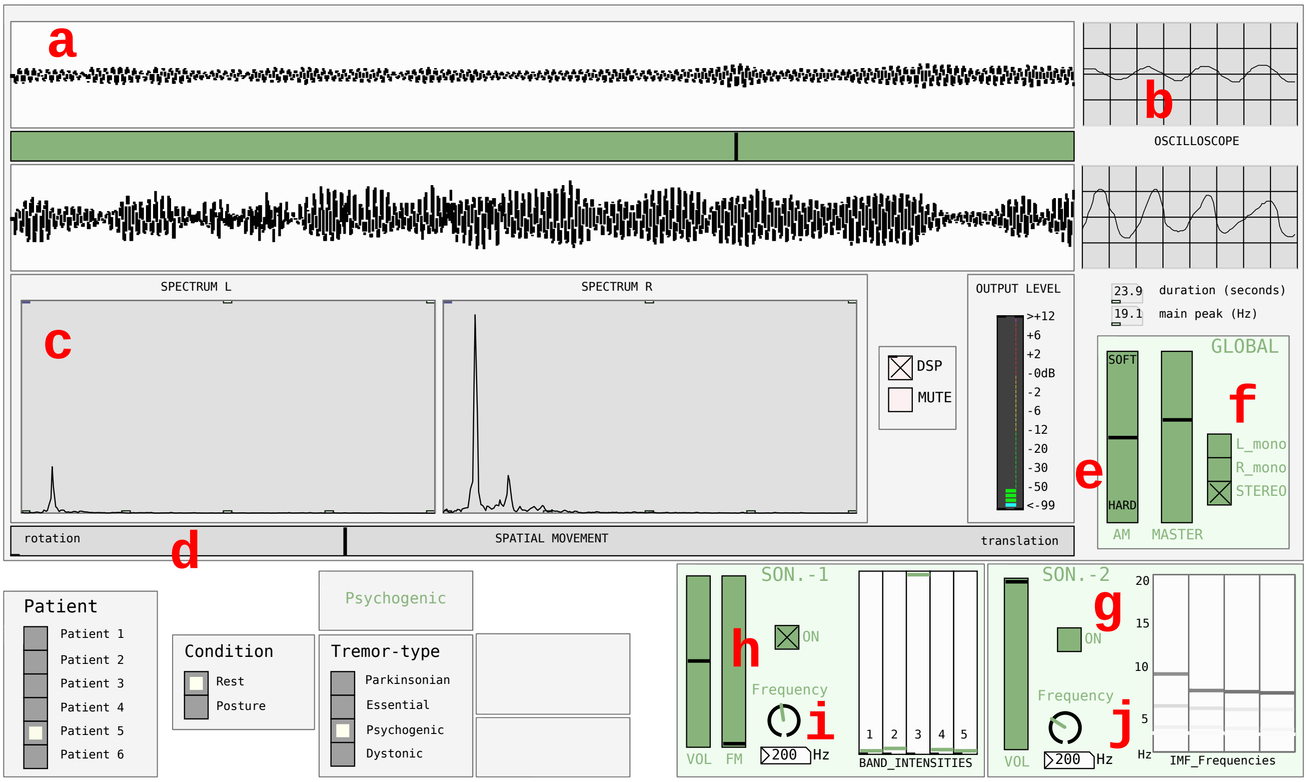


Figure 1. Graphical user interface (here in training mode). Annotations in red letters. Visual feedback: a) waveform, b) oscilloscope, c) FFT spectrum, each for left and right channel, d) ratio between rotational and translational movement. Global parameters: e) smoothing of the AM modulator, f) mono/stereo and left/right channel switch, g) sonification switch. Vocoder sonification: h) FM modulation, i) fundamental frequency. EMD sonification: j) frequency factor.

lected 18 patients of each type to form the reduced dataset of 18×4 = 72 patients.

Test participants for the extended study were recruited from an expert listening panel [18, 19], a group of musicians and sound engineers with experience in listening tests. Despite the target audience being neurologists, trained listeners have been chosen to ensure a best-case scenario and hence a more fair comparison of the results with the currently achieved diagnostic accuracy through visual and computer-aided ex-post analysis methods. Neurologists can acquire these abilities provided that they obtain appropriate ear training. The sonification was presented with closed headphones. Test participants were allowed to take handwritten notes throughout the experiments.

We carried out two experiments which are discussed below: a 4AFC identification task (Sec. 3.1) similar to the pilot study (cf., [6]) and a triangle test (also called 3AFC oddity task) which is described in Sec. 3.2.

3.1 Experiment 1: training and identification task

The basic design of Experiment 1 was similar to the pilot study [6]. It further addresses learning effects over several training sessions. An additional focus of this experiment was the interactive use of sonification parameters. The goal was to verify the results of the pilot study with a larger number of patients and tremor types.

3.1.1 User Interface

Apart from the sonic representation of the tremor data, a simple visualization is provided (see Fig. 1). It shows various visual information, such as waveform view, oscillogram, level meter, FFT spectrum, ratio between rotational and translational movement strength as well as band intensities and IMF frequencies for the individual sonifications.

Further, apart from standard controls such as volume, the interface features interactive access to a selection of sonification parameters. Globally for both sonifications, the smoothing of the AM modulator signal as well as stereo/ mono playback can be controlled. In addition, each sonification allows individual access to oscillator frequency (base frequency for Vocoder sonification, multiplicative factor for EMD sonification) and dedicated gains for left and right arm sensors. Another slider provides control of the FM index for the Vocoder sonification.

3.1.2 Procedure

In Experiment 1, participants had to accomplish three sessions (S1–S3) of one hour on different days with approximately one week of pause in between. Each session started with approx. 30 minutes of free training where participants could freely listen to the sonifications of 8 patients per tremor type (24 patients in total) with disclosed diagnosis.

Participants were able to decide by themselves when they felt to be ready for the next part of the session, in which they had to identify the disease of 24 patients (6 per tre-

	Session 1	Session 2	Session 3	Overall (S1 and S3)
Percent correct answers	31.5 (10.2)	39.1 (7.3)	31.5 (10.0)	31.5 (9.9)
Confidence Interval CI95	26.9 − 36.4	34.2 − 44.1	26.9 − 36.4	28.2 − 34.9
Confidence (from 1 to 100)	38.8 (24.2)	43.3 (23.5)	39.9 (26.9)	39.4 (25.6)
Response time (seconds)	34.6 (25.2)	41.9 (27.1)	42.1 (33.9)	38.4 (30.1)

Table 1. Overview of the results from Experiment 1. Values in parentheses describe one standard deviation.

mor type) by using the interactive sonification interface. The task followed the same procedure as in the pilot study. Feedback was given after each trial.

For each participant individually, the dataset was split randomly into 3 subsets of 24 patients (6 per tremor type). These 3 subsets were presented in the training and testing parts of the 3 sessions. The presentation order of patients within each session and within each mode (training/ testing) was randomized across participants. In the first and in the last session, training was performed with subset A, while a completely new subset (B/C) was introduced for the testing part. In the second session, both training and testing were performed with the same subset (B):

	S1	S2	S3
Training	A	B	A
Testing	B	B	C

S1 and S3 are seen as a realistic scenario where known patients' characteristics were applied to unknown patients. In S2, the ability of memorizing specific patients' characteristics was tested, comparable to the procedure in the pilot study.

During the testing part of each session, participants were asked to submit a diagnosis (4AFC) and judge their confidence (values from 1 to 100) for each of the 24 patients. Feedback (reference diagnosis) was given after each trial. Sonification-specific parameters as well as the condition (rest/posture) could be additionally controlled by a motorized MIDI controller (Behringer BCF-2000).

3.1.3 Results

Overall results of the identification task are shown in Tab. 1. On average, the test participants achieved 31.5% correct answers for S1 and S3, while S2 led to 39.1 % correct answers. Confidence intervals in Tab. 1 are based on the inverse Beta CDF. The total percent correct answers lead to a discrimination index d' of 0.26 for S1/S3 and 0.49 for S2 [20].

The primary test results (percent correct answers) were analyzed by using a binomial test with one variable "disease" (4 levels), assuming a constant hit rate of $1/4$, sample size of 384 (24 patients × 16 participants), and significance level of p=0.05. For all three sessions, the achieved percent correct answers are significantly higher than chance (p=0.002 for S1/S3, p<0.001 for S2). The overall results of the first and third session are identical and therefore considered equivalent.

For further analysis of the results, contingency tables were created for the individual sessions (Tab. 2a–2c) as well as for the pooled results of S1 and S3 (Tab. 2d). The tremor types Parkinsonian, Essential, Psychogenic, and Dystonic, are referred to as Par, Ess, Psy, and Dys.

The higher percent correct responses in S2 compared to S1 and S3 is further supported by the higher contingency coefficient[4] (0.32 vs. 0.22 and 0.25), which means that there is a stronger relation between reference diagnosis and submitted diagnosis for S2 compared to both other sessions.

The pooled results of S1 and S3 (Tab. 2d) reveal a high confusion between Parkisonian and Psychogenic tremor (31.8 % Par falsely assigned to Psy, 28.1 % Psy falsely assigned to Par). The lowest false positive rate appeared between Parkinsonian and Dystonic tremor (16.1 % Par falsely assigned to Dys, 18.8 % Dys falsely assigned to Par).

A binomial test as above was performed on the results per tremor type. When looking at the pooled results of S1 and S3, only the percent correct identifications of Parkinsonian, Psychogenic, and Dystonic tremor were significantly higher than chance (Par: p=0.014, Psy: p=0.042, Dys: p=0.004), while Essential tremor could not be identified (p=0.107). However, the differences in sensitivity (Par: 0.32, Ess: 0.29, Psy: 0.31, Dys: 0.34) and F-measure (Par: 0.38, Ess: 0.36, Psy: 0.33, Dys: 0.42) are only small. The difference in percent correct responses between the individual tremor types therefore seems marginal.

We did not find any significant clusterization of test participants based on their individual contingency tables. Also, the results of those participants who primarily used the vocoder-based sonification did not significantly differ from the results of those wo preferred the EMD-based sonification (9 participants vs. 7 participants, 32.3 % vs. 30.4 % correct answers as average of S1 and S3). However, there were five top performers who achieved at least 33.3 % correct answers in every session (Average of S1 and S3: 36.6 % correct answers, ranging from 33.3 % to 47.7 %).

On average (pooled over S1 and S3), participants took 38.4 s (SD=30.1 s) to complete one trial and reported a confidence of 39.4 (SD=25.6).

3.1.4 Interactive use of sonification parameters

For the slider- or rotary-based controls (AM smoothing, Vocoder sonification base frequency and FM index, EMD sonification frequency factor), we observed that participants found their preferred values already within the first training session. These parameters were rarely changed during diagnosis.

The sonification switch also exhibited convergent behavior for all but one participant who was continuously switch-

[4] Pearsons's contingency coefficient: $K = \sqrt{\frac{Chi^2}{N+Chi^2}}$, where N is the sample size.

D R	Par	Ess	Psy	Dys
Par	**32.3**	20.8	31.3	15.6
Ess	28.1	**22.9**	25.0	24.0
Psy	28.1	20.8	**34.4**	16.7
Dys	24.0	22.9	16.7	**36.5**
Sum	112.5	87.5	107.3	92.7

(a) Session 1. $K = 0.22$.

D R	Par	Ess	Psy	Dys
Par	**36.6**	16.7	21.9	21.9
Ess	21.9	**38.5**	19.8	19.8
Psy	18.8	18.8	**40.6**	21.9
Dys	13.5	21.9	27.1	**37.5**
Sum	93.8	95.8	109.4	101.0

(b) Session 2. $K = 0.32$.

D R	Par	Ess	Psy	Dys
Par	**32.3**	18.8	32.3	16.7
Ess	14.6	**35.4**	30.2	19.8
Psy	28.1	18.8	**27.1**	26.0
Dys	13.5	21.9	33.3	**31.3**
Sum	88.5	94.8	122.9	93.8

(c) Session 3. $K = 0.25$.

D R	Par	Ess	Psy	Dys
Par	**32.3**	19.8	31.8	16.1
Ess	21.4	**29.2**	27.6	21.9
Psy	28.1	19.8	**30.7**	21.4
Dys	18.8	22.4	25.0	**33.9**
Sum	100.5	91.1	115.1	93.2

(d) Sessions 1 & 3. $K = 0.19$.

Table 2. Contingency tables for the three sessions and the pooled results of S1 and S3. Values describe % of submitted diagnoses D. R is the reference diagnosis. Correct answers (main diagonal) are highlighted. K is Pearson's contingency coefficient.

ing between sonifications throughout the whole experiment. The average amount of sonification switches per trial decreased from 0.57 switches in S1 to 0.27 in S2 and 0.08 in S3. Both other toggles (mono/stereo and rest/posture) never converged during the experiment (1.22 and 1.46 switches per trial in S1 vs. 1.42 and 2.21 in S3, respectively). Even in S3, 7 participants regularly switched to mono, and 14 participants regularly switched between rest and posture position.

3.1.5 Discussion

The quick convergence of the slider- and rotary-based parameters (already during the first training session) indicates that they did not contribute to the test participants' judgments of tremor diseases. During informal interviews, test participants mentioned that the base frequency for the Vocoder sonification and the frequency factor for the EMD sonification had only minor effect on the information conveyance and were only changed in respect to personal preference. Also frequency modulation for the Vocoder sonification did not add additional information but was often applied for diversification of the otherwise relatively monotonic sounds. For the smoothing factor of the amplitude modulation, however, participants had to find a compromise: while less smoothing leads to higher rhythmic resolution, the stronger transients implicate bandwidth expansion and therefore deteriorated perception of frequency changes. While fixed default values for the mentioned parameters might work for most users, an optional parametrization for expert users is assumed to be advantageous. The interactive switch between sonifications has been used regularly by at least one test participant, which further emphasizes its necessity.

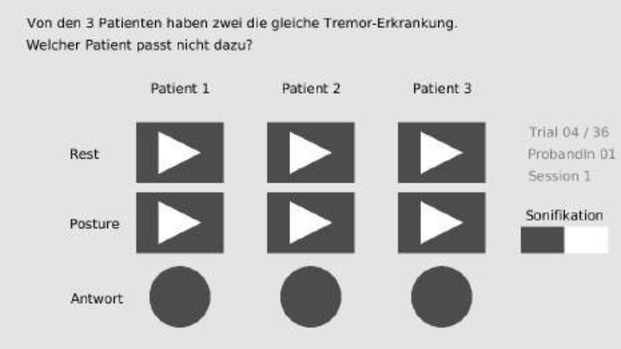
The stereo/mono (connected with left/right switch) and rest/posture switch were used excessively by most test participants. During informal interviews, many test participants mentioned that a comparison between left and right arm sensor as well as between rest and posture condition revealed critical information for tremor type identification. These parameters are therefore regarded as essential for acoustic tremor analysis.

The discrimination between 4 tremor types by using the interactive sonification interface seems to be excessively demanding (31.5 % correct diagnoses). According to their hand-written notes as well as informal interviews, most of the test participants were focused on finding specific sound characteristics that are unique to a specific tremor disease. However, there were outlying patients in all tremor categories which led to confusion. This observation is also reflected in the submitted confidence values which did not correlate with the correctness of the submitted diagnoses.

The better performance in the second session shows that the test participants were able to remember specific patients' sound characteristics and corresponding reference diagnosis (39.1 % correct responses in S2 vs. 31.5 % in S1/S3). This also explains the promising results of the pilot study: in both pilot study and S2 of Experiment 1, the training was performed with the same set of patients. In real life, however, a diagnosis is only needed for unknown patients. In the more realistic conditions of S1 and S3, the test participants could not rely on patient-specific sound characteristics, but only on more general tremor-specific structures.

There was no improvement after training (S1 vs. S3). We assume that the training parts of the second and third sessions were needed to recapitulate the individual tremor types' sound characteristics. Further, we noticed that the new set of patients in S2 did not always match the participants' notes, which forced them to start again from scratch. We suppose that training time was not sufficient for finding the possibly very subtle differences between different tremori.

Both sonifications were used equally often and provided similar results. This finding is consistent with the pilot study and shows once more that an optional choice could be important in order to adapt to the users' individual preferences.



Von den 3 Patienten haben zwei die gleiche Tremor-Erkrankung.
Welcher Patient passt nicht dazu?

Figure 2. Graphical user interface for Experiment 2. Colors are inverted and converted to grayscale for this article.

A further interpretation of participants' confidence and response time is not possible due to the high spread.

The results of Experiment 1 suggest that an absolute identification of tremor diseases with the proposed interface might be too difficult. However, the contingency table in Tab. 2d shows that some pairs of tremori were less often confused with each other than other pairs. In order to verify these pairings, we performed a second experiment.

3.2 Experiment 2: triangle test

After having accomplished the training and absolute diagnosis task of Experiment 1, the same 16 participants performed a discrimination task comprising of 72 trials which were equally split in two sessions of approx. one hour each, performed on different days.

3.2.1 Procedure

Before the actual test, all participants performed a mandatory training with the interactive sonification interface in order to recapitulate the different tremor characteristics.

In a triangle test setup (3AFC oddity task), only a simple graphical user interface without visual tools was shown (see Fig. 2). Sonifications of three different patients were presented: two of the same tremor type and one of a different type. For each triple of patients, participants had to indicate the odd, i.e., the patient whose tremor type differed from the others'. After each trial, participants obtained feedback consisting of the reference diagnosis for all three patients, with the correct answer highlighted.

For each participant and session individually, the patients of the trials were chosen randomly so that every patient appeared once as the odd stimulus and twice inside the pair of similar stimuli. The three patients per trial as well as the 36 trials per session were presented in random order.

Participants could freely listen to the sonifications of all three patients in rest as well as in posture condition. There was no restriction in time. Switching between Vocoder and EMD sonification was still possible. Sound was always presented in stereo in order to limit the possibilities for interaction to a minimum.

3.2.2 Results

The overall results of Experiment 2 are shown in Tab. 3 for the 6 different pairs of tremor types individually. On average, test participants achieved $41.2\,\%$ correct judgments, yielding a sensitivity index d' of 0.95 (based on [21]). Test participants achieved above-average results only when discriminating Dystonic tremor from Parkinsonian or Psychogenic tremor (45.8 and $49.5\,\%$ correct, respectively).

Type pair	% correct	d'	response time
Par vs. Ess	39.6	0.88	59.7
Par vs. Psy	38.5	0.81	51.5
Par vs. Dys	45.8	1.25	58.4
Ess vs. Psy	38.0	0.73	50.3
Ess vs. Dys	35.9	0.55	54.5
Psy vs. Dys	49.5	1.41	52.7
Overall	41.2	0.95	54.5

Table 3. Overview of the results per pair of tremor types in Experiment 2: average percent correct responses, sensitivity index d' [21], and response time in seconds.

For statistical analysis of the results, a one-tailed binomial test was performed for each pair of tremor types (sample size $n{=}192$ (pooled over all subjects), chance level $p{=}1/3$, significance level: $p{=}0.05$). According to the binomial test there was a highly significant perceptual difference between Parkinsonian and Dystonic as well as between Psychogenic and Dystonic tremor ($p{<}0.001$ for both pairs). A marginally significant difference between Parkinsonian and Essential tremor was found ($p{=}0.040$). Between the other three pairs of tremor types no significant perceptual difference could be found ($p{=}0.074$ for Par/Psy, $p{=}0.097$ for Ess/Psy, and $p{=}0.244$ for Ess/Dys).

We observed that participants usually chose their preferred sonification at the beginning and did not change it anymore. User interaction is therefore not further investigated.

The average response time (time to accomplish one trial) was 54.5 seconds ($SD{=}35.0$).

In order to examine similarities between individual test participants, we performed a hierarchical cluster analysis. Individually for each participant, the total percent correct responses for the pairs of tremor types form a 6-dimensional vector. Based on these vectors and Ward's method of minimum inner squared euclidean distance within clusters [22], test participants divide equally into two groups of 8. Results are shown in Tab. 4. While Cluster 1 inhibits only small differences in percent correct responses between tremor pairs, the participants of Cluster 2 were exceptionally good at discriminating Dystonic tremor from Parkinsonian and Psychogenic tremor ($62.5\,\%$ and $54.2\,\%$, respectively). Interestingly, in Cluster 1, all participants except two preferred the EMD sonification, while in Cluster 2 all participants except one preferred the Vocoder sonification.

Differences in overall percent correct responses between participants grouped by cluster or sonification preference, however, were not significant.

Five of the 16 participants (see "Top 5" in Tab. 4) reached a level of above $45\,\%$ correct responses (on average: $47.8\,\%$, ranging from $45.8\,\%$ to $50.0\,\%$). Three of those used the Vocoder Sonification while two worked with the EMD Sonification.

Type pair	Cluster 1	Cluster 2	Top 5
Par vs. Ess	44.8	34.4	35.0
Par vs. Psy	38.5	38.5	50.0
Par vs. Dys	29.2	62.5	50.0
Ess vs. Psy	39.6	36.5	48.3
Ess vs. Dys	42.7	29.2	43.3
Psy vs. Dys	44.8	54.2	60.0
Average	39.9	42.5	47.8

Table 4. Average results per pair of tremor types in Experiment 2 for the two main clusters and the 5 best performing participants. Values describe % correct answers.

3.2.3 Discussion

The overall discrimination index d'=0.95 in Experiment 2 shows that the oddity task was indeed easier than the identification task of Experiment 1 (d'=0.26). However, overall percent correct responses are still too low for direct applications in the medical context.

A cluster analysis on the basis of the test participants' results leads to an almost perfect division into the users of the different sonifications. While the Vocoder sonification seems to provide similar discrimination performance for all tremor pairs, users of the EMD sonification achieved extraordinary results for the pairs Par/Dys and Psy/Dys (at the expense of lower percent correct responses for the other pairs). Each sonification might emphasize different aspects of the observed tremori. If those aspects were known, participants might be able to combine both sonifications in order to improve their sensitivity. The interactive change between both sonifications therefore seems reasonable and might facilitate the construction of a coherent picture based on the observed phenomena in order to make informed decisions.

The five top performers of Experiment 2 (Tab. 4) also achieved high identification rates in Experiment 1 (on average: 33.8 % correct, ranging from 31.3 % to 37.5 %, two of them among the Top 5). This leads to the assumption that at least some test participants have found consistent tremor-specific sound characteristics and were able to apply these complex models in an identification task as well as in a discrimination task.

4. CONCLUSIONS AND OUTLOOK

In the presented study, we evaluated tremor disease identification and discrimination by means of interactive sonification.

In Experiment 1, test participants were asked to indicate the tremor type of unknown patients by analyzing pre-recorded movement data with the interactive sonification interface. Average percent correct diagnoses were significantly above chance for patients with Parkinsonian, Psychogenic, and Dystonic tremor, but not for Essential tremor. Overall sensitivity index d' was 0.26.

In Experiment 2, participants were able to distinguish Parkinsonian tremor from Essential and Dystonic tremor with sensitivity significantly above guessing rate. A significant perceptual difference was also shown between Dystonic and Psychogenic tremor. Overall d' was 0.95.

During the first experiment, participants were allowed to manipulate critical sonification parameters. All test participants found their preferred sonification and corresponding parameters quickly. Still, the permanent comparison between left and right arm sensor as well as between rest and posture condition provided useful information. The interactive change of these parameters facilitates the construction of a coherent image of the observed tremor and allows informed decision making.

The proposed sonification interface is not meant to replace currently available diagnostic tools, but rather to complement them. The fact that both sonifications led to different results in the triangle test lets us conjecture that they could supplement each other in a beneficial way, especially if additional knowledge from other diagnostic tools is available. A combination and cross-checking of different sources of information is essential for an efficient and correct diagnosis.

It is obvious that both sonifications as well as the corresponding user interface need to be substantially revised. This revision could be based on the test participants' qualitative feedback which we collected. Informal interviews with test participants revealed that the offered sonification parameters might not be sufficient. One of the top performing participants complained that the first IMF can become obtrusive and often masks the important fine structure of the higher IMFs; however, at the same time, it also contains critical information and can therefore not be omitted. Individual volume control for the IMFs may therefore an option for future improvements.

Further, we observed that some test participants successfully used the spatial movement visualization for decision making: while Parkinsonian and Dystonic tremor showed almost only rotational movement, Essential and Psychogenic tremor sometimes caused sudden switches towards a translational movement. At this stage, however, the ratio between rotational and translational movement is not conveyed by the sonification. A prior approach mapping this movement quality to a chorus effect (see [6]) did not work due to the parameter's transient behavior. A promising option might be a mapping of the translation/rotation-ratio to relative pitch.

Finally, we think that the collected strategies of the test participants could be brought in correspondence with Music Information Retrieval (MIR) analysis descriptors applied to sonifications and raw movement data. Thus, the sonification qualities which have proven to be perceptually relevant in tremor segregation could be enhanced in the next development steps. In general, the re-design of the developed sonifications could be informed by the findings of this combined qualitative and quantitative analysis.

5. ACKNOWLEDGMENTS

We would like to thank Petra Schwingenschuh and the Medical University of Graz, who kindly provided us with the clinical tremor data and thus made this research possible.

6. REFERENCES

[1] K. T. Wyne, "A comprehensive review of tremor," *JAAPA*, vol. 18, no. 12, pp. 43–50, 12 2005.

[2] G. Deuschl, P. Bain, and M. Brin, "Consensus statement of the Movement Disorder Society on Tremor. Ad Hoc Scientific Committee," *Mov. Disord.*, vol. 13 Suppl 3, pp. 2–23, 1998.

[3] M. Schedel, D. Weymouth, T. Pinkhasov, J. Loomis, I. B. Morris, E. Vasudevan, and L. Muratori, "Interactive sonification of gait: Realtime biofeedback for people with parkinson's disease," in *5th Interactive Sonification Workshop (ISon)*, 12 2016.

[4] A.-M. Gorgas, L. Schön, R. Dlapka, J. Doppler, M. Iber, C. Gradl, A. Kiselka, T. Siragy, and B. Horsak, *Short-Term Effects of Real-Time Auditory Display (Sonification) on Gait Parameters in People with Parkinsons' Disease – A Pilot Study*, 2017, pp. 855–859.

[5] D. Pirrò, A. Wankhammer, P. Schwingenschuh, R. Höldrich, and A. Sontacchi, "Acoustic interface for tremor analysis," in *Proc. International Conference on Auditory Display (ICAD)*, 02 2012.

[6] M. Weger, D. Pirrò, A. Wankhammer, and R. Höldrich, "Discrimination of tremor diseases by interactive sonification," in *5th Interactive Sonification Workshop (ISon)*, 12 2016.

[7] IEM. (2017) Acoustic interface for tremor analysis. [Online]. Available: http://tremor.iem.at

[8] J. Shlens, "A tutorial on principal component analysis," in *Systems Neurobiology Laboratory, Salk Institute for Biological Studies*, 2005.

[9] N. E. Huang, Z. Shen, S. R. Long, M. C. Wu, H. H. Shih, Q. Zheng, N.-C. Yen, C. C. Tung, and H. H. Liu, "The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis," *Proc. of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 454, no. 1971, pp. 903–995, 1998.

[10] A. Zeiler, R. Faltermeier, I. R. Keck, A. M. Tomé, C. G. Puntonet, and E. W. Lang, "Empirical mode decomposition - an introduction," in *International Joint Conference on Neural Networks (IJCNN)*, 7 2010, pp. 1–8.

[11] M. C. Peel, G. E. Amirthanathan, G. G. S. Pegram, T. A. Mcmahon, and F. H. S. Chiew, "Issues with the Application of Empirical Mode Decomposition Analysis," in *International Congress on Modelling and Simulation (MODSIM)*, 12 2005, pp. 170–176.

[12] R. T. Rato, M. D. Ortigueira, and A. G. Batista, "On the HHT, its problems, and some solutions," *Mechanical Systems and Signal Processing*, vol. 22, pp. 1374–1394, 8 2008.

[13] N. E. Huang and S. S. Shen, Eds., *Hilbert-Huang Transform And Its Applications*. World Scientific, 2005.

[14] S. Ayache, T. Al-ani, and J.-P. Lefaucheur, "Distinction between essential and physiological tremor using hilbert-huang transform," *Neurophysiologie Clinique/Clinical Neurophysiology*, vol. 44, no. 2, pp. 203–212, 2014.

[15] S. S. Ayache, T. Al-ani, W. H. Farhat, H. G. Zouari, A. Creange, and J. P. Lefaucheur, "Analysis of tremor in multiple sclerosis using Hilbert-Huang Transform," *Neurophysiologie Clinique/Clinical Neurophysiology*, vol. 45, no. 6, pp. 475–484, 12 2015.

[16] I. Carpinella, D. Cattaneo, and M. Ferrarin, "Hilbert-Huang transform based instrumental assessment of intention tremor in multiple sclerosis," *J Neural Eng*, vol. 12, no. 4, 8 2015.

[17] N. E. Huang, Z. Wu, S. R. Long, K. C. Arnold, X. Chen, and K. Blank, "On instantaneous frequency," *Advances in Adaptive Data Analysis*, vol. 1, no. 2, pp. 177–229, 2009.

[18] R. Höldrich, H. Pomberger, and A. Sontacchi, "Recruiting and evaluation process of an expert listening panel," in *Fortschritte der Akustik (NAG/DAGA 2009)*, B. Rinus, Ed., Rotterdam (Niederlande), 03 2009.

[19] M. Frank and A. Sontacchi, "Performance review of an expert listening panel," in *Fortschritte der Akustik (DAGA 2012)*, H. Hanselka, Ed., Berlin (Deutschland), 09 2012.

[20] M. J. Hacker and R. Ratcliff, "A revised table of d' for m-alternative forced choice," *Perception & Psychophysics*, vol. 26, no. 2, pp. 168–170, 1979.

[21] B. J. Craven, "A table of d' for m-alternative odd-man-out forced-choice procedures," *Perception & Psychophysics*, vol. 51, no. 4, pp. 379–385, 1992.

[22] J. H. W. Jr., "Hierarchical grouping to optimize an objective function," *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 236–244, 1963.

AN EXPLORATORY STUDY ON THE EFFECT OF AUDITORY FEEDBACK ON GAZE BEHAVIOR IN A VIRTUAL THROWING TASK WITH AND WITHOUT HAPTIC FEEDBACK

Emma Frid, Roberto Bresin, Eva-Lotta Sallnäs Pysander
KTH Royal Institute of Technology
{emmafrid, roberto, evalotta}@kth.se

Jonas Moll
Uppsala University
jonas.moll@it.uu.se

ABSTRACT

This paper presents findings from an exploratory study on the effect of auditory feedback on gaze behavior. A total of 20 participants took part in an experiment where the task was to throw a virtual ball into a goal in different conditions: visual only, audiovisual, visuohaptic and audiovisuohaptic. Two different sound models were compared in the audio conditions. Analysis of eye tracking metrics indicated large inter-subject variability; difference between subjects was greater than difference between feedback conditions. No significant effect of condition could be observed, but clusters of similar behaviors were identified. Some of the participants' gaze behaviors appeared to have been affected by the presence of auditory feedback, but the effect of sound model was not consistent across subjects. We discuss individual behaviors and illustrate gaze behavior through sonification of gaze trajectories. Findings from this study raise intriguing questions that motivate future large-scale studies on the effect of auditory feedback on gaze behavior.

1. INTRODUCTION

Sound has been found to influence visual perception [1, 2] and affect attention [3]. It has been suggested that sound effects can be used to increase selective attention and inferential recognition among children [4] and that sound can facilitate visual learning [5]. The above-mentioned findings highlight the potential benefits of integrating auditory feedback in tasks requiring a certain level of visual focus. Despite the possible benefits of adding sound to such tasks, few studies have focused on the effect of auditory feedback on gaze behavior when interacting with a multimodal interface. The body of research related to this topic mainly involves studies on the effect of auditory feedback when watching videos. It has been found that different kinds of sounds influence gaze differently [6, 7]. In one study on soundtracks in videos, it was concluded that sound may influence eye position, fixation duration and saccade amplitude, but that the effect of sound is not necessarily constant over time [8]. In another study [9], it was found that the influence of audio depends on the consistency between the visual and audio signals: if the salient objects from the visual perspective are not consistent with the salient objects from the audio perspective, audio will influence visual attention. Otherwise, there is little influence on visual attention.

There has been some research on sonification of gaze behavior, e.g. [10], focusing on how sonification can be used to guide visual attention and [11], focusing on sonification for eye movement control. In [11], the effects of sonification on performance and learning were evaluated using an interface that sonified eye movements. Very heterogeneous behaviors were observed among subjects and the effect of movement sonification was therefore evaluated on an individual level. A similar approach, based on analysis on subject level, was used in [10].

In the studies referenced above, eye tracking technology was used to record gaze behavior. Eye tracking is a technique in which an individual's eye movements are measured in order to detect where a person is looking at a specific time [12]. Common eye tracking metrics include measures of fixations, such as e.g. total fixation duration and number of fixations. Fixations are moments when the eyes are relatively stationary due to information processing. Typically, fixations are correlated with attention, but interpretation of fixation metrics may differ depending on the specific nature of the study. Based on top-down cognitive theory, longer fixation durations in a specific region of an interface would reflect a participant's difficulty in interpretation for this particular area. Fixation duration is believed to relate to the cognitive effort expended [13, 14]. However, long fixation duration could also indicate that a specific area is more engaging in some way [15]. A high number of fixations in a particular area could also imply that the region is more noticeable or important than other regions [12].

In this paper we present main findings from a study on the distribution of visual attention in a multimodal single-user application comparing different modalities. We investigated the effect of auditory feedback on visual attention and gaze behavior. An experiment was performed with 20 participants. The participants were asked to throw a virtual ball into a goal in different feedback conditions. Eye tracking methodologies were used to investigate where the visual sensory information was acquired when performing the task. We evaluated the effect of auditory feedback on eye tracking metrics and gaze behavior, both when haptic force feedback was provided and when no haptic force feedback feedback was provided. The results presented in this paper are a subset of the results from the performed study. See [16] for a discussion about several other measures.

2. METHOD

One of the main aims of the study was to investigate if, and how, auditory feedback alters gaze behavior and distribution of visual attention in a nonhaptic versus haptic task. More dimensions are covered in [16]. The user was asked to throw a virtual ball into a goal, see Fig. 1. A throwing gesture was chosen since this is an intuitive and simple movement task that requires attention to shift between an object and a target. This gesture could also be intuitively sonified. We define the movement performed in this study as a "virtual throwing gesture", since the haptic device that was used has limited possibilities in terms of affording a "real" throwing gesture.

2.1 Hypothesis

We hypothesized that continuous auditory feedback effectively represents temporal information in such a manner that a shift in attention is enabled in a virtual (visual) throwing task. More specifically, our hypothesis was that *the use of interactive movement sonification will enable a user to shift visual focus from an actual interaction point (the location of the cursor) to a more distant target point in a virtual 3D environment.* Sonification can provide meaningful information about a movement. In conditions involving sonification, the user will not be required to focus as much on the ball, since information about the performed movement is communicated directly through sound. This will enable visual focus to shift from the ball, to the goal. Moreover, we hypothesized that *gaze behavior will be differently affected when haptic force feedback (adding a sense of weight and inertia) is simultaneously presented with movement sonification*.

2.2 Participants

A total of 20 participants took part in the experiment. All participants were first year students at KTH Royal Institute of Technology (11 M, 9 F; age=19-25 years, mean = 20.4 years). None of the participants reported having any hearing deficiencies or reduced touch sensitivity. All participants reported that they had normal or corrected-to-normal vision. Six of the participants normally wore glasses; two of them wore their glasses during the test sessions and two of them used contact lenses. None of the participants reported that they had any previous experience of using force feedback devices. All participants signed a consent form prior to taking part in the experiment. Six participants were disregarded from the analysis since the level of usable gaze data was below 80% [1]. Yet another participant was disregarded since the headphones accidentally switched off during the experiment. The final sample size for analysis was n=13 (8 M, 5 F; age=19-24 years, mean = 20.8 years).

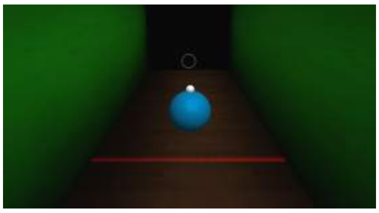


Figure 1. Screenshot of the graphical interface.

2.3 Apparatus

2.3.1 Technical Setup

The technical setup can be seen in Fig. 2. One desktop computer was used to run the haptic interface and the eye tracking application; one laptop computer was used for sound generation. The 3D interface was designed to allow the user to grasp a ball and to throw it into a circular goal area. A SensAbleTM Phantom® Desktop haptic device [2] was used to provide haptic feedback. This haptic device has a pen-like stylus that is attached to a robotic arm. The stylus can be used to haptically explore objects in virtual 3D environments and to provide force feedback. Weight and inertia can also be simulated. In our experiment, a button on the stylus was used to activate the function of grasping an object, enabling the user to lift and move the virtual ball. The 3D application, shown in Fig. 1, was based on the haptic software library Chai3D [17] and developed in C++.

Eye tracking data was captured at a sampling rate of 60 Hz using a commercial X2-60 eye-tracker from Tobii Technology [3] connected to an external processing unit. The software Tobii Studio was used for calibration and analysis of eye tracking data. Two interactive sonification models were implemented using Max/MSP [4]. Communication between Max/MSP and the 3D application was made possible via Open Sound Control (OSC) [18]. A pair of Sennheiser RS 220 headphones was used to provide auditory feedback during the experiment.

2.3.2 Sonification

For conditions involving auditory feedback, the movements of the virtual ball were continuously sonified. Two different sound models were used for sonification of the throwing gesture, i.e. when aiming towards the goal. The first sound model (from now on referred to as S1) was based on a physical model of friction, readily available from the Sound Design Toolkit [5] [19] (SDT). The second sound model (from now on referred to as S2) was obtained by filtering pink noise using the Max object `[biquad]` with a low-pass filter setting. We opted for these particular sound

[1] 100% in this context would imply that gaze data for both eyes were found throughout the entire recording.

[2] `http://www.dentsable.com/haptic-phantom-desktop.htm`

[3] `www.tobiipro.com`

[4] `https://cycling74.com/`

[5] `http://soundobject.org/SDT/`

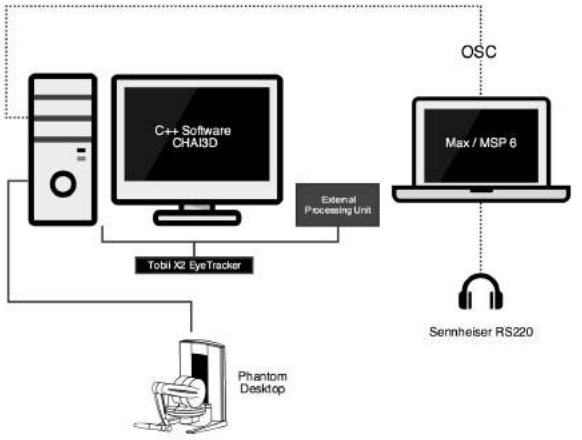


Figure 2. Technical setup.

models since they were hypothesized to be very perceptually different; S1 produced creaking sounds whereas S2 produced swishing sounds [6] . For sound model S1, x-,y- and z-positions were mapped to panning, frequency, and rubbing force, respectively. For S2, x-,y- and z-positions were mapped to panning, frequency and comb-filter characteristics, respectively. Velocity was mapped to amplitude for both sound models.

Different impact sounds from the Sound Design Toolkit (SDT) were used for the two sound models, in order to simulate different weights of the ball. These sounds were included to enable evaluation of the overall effect of auditory feedback (i.e. not only the effect of movement sonification). Parameters were chosen in such a manner that a sound reminiscent of a heavy rubber ball was used for bouncing events in S1, whereas the bouncing effect was tuned to remind more of the impact sound created by a very light table tennis ball (a ping pong ball) for S2. A set of interaction sounds were also included in the application in order to communicate occurrence of the following events: scoring a goal (MIDI sequence, increasing pitch), missing the goal (MIDI sequence, decreasing pitch), hitting a wall (SDT impact sound model used to create a sound reminding of a dissonant bell) and successfully grasping the ball (filtered noise with increasing frequency, producing a sweeping sound that finishes with a short “click”).

2.4 Procedure

We opted for a within-participant design in which repeated measures were done on each participant. Participants performed the throwing task in six different feedback conditions. The following conditions were included in the experiment: visual only, i.e. visuo (V), audiovisual with a creaking sound (AV1), audiovisual with a swishing sound (AV2), visuohaptic (VH), audiovisuohaptic with a creaking sound (AVH1) and audiovisuohaptic with a swishing sound (AVH2). A randomized order of the six conditions was presented to each participant.

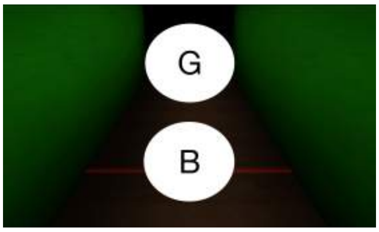


Figure 3. Definition of areas of interest. G stands for the goal AOI and B for the ball AOI.

The experiment consisted of the following sub-parts: (1) general introduction and calibration of the system, (2) three condition sessions, (3) a 10 minute break, (4) three condition sessions and (5) concluding interview. Every condition was preceded by a practice trial (minimum 2 and maximum 4 minutes) in which participants got the chance to familiarize themselves with solving the task under the current feedback condition. The practice trials were included in order to reduce the risk of potential learning effects. The task was to throw the ball into the goal until 15 successful hits had been reached. Each experiment lasted about 1.5 hours. The concluding interview consisted of a discussion about aiming strategies and the visual focus on different areas of the screen.

2.5 Analysis

A common method in analysis of eye tracking data is to initially define areas that are of particular interest for the research hypothesis; Areas of Interest (AOIs). In this experiment, three AOIs were defined: one for the goal area (G), one for the ball area (B), and one for the entire screen. The G and B AOIs are displayed in Fig. 3. These elliptical AOIs were defined after visual inspection of heat maps and automatic generation of clusters of fixations, using the Tobii Studio software. The two AOIs had the same size.

The following metrics were computed for the AOIs: *total fixation duration* and *total fixation count*. All calculations were done on 15 throwing gestures. This was done in order to obtain a measure that was comparable despite the fact that task duration and performance varied substantially between participants (the total number of throwing gestures varied from 15 to 66). No consistent patterns in terms of correlations between error rates (number of unsuccessful attempts to score a goal for a total of 15 achieved hits) and eye tracking metrics could be observed [7] . For a more detailed discussion on task performance, see [16].

In the following section we provide descriptive statistics for the two eye tracking metrics, but without statistically significant conclusions when averaging the subjects’ values [8] . Participants exhibited large inter-individual variability for both the total fixation duration and the total fixation count metric. Some general trends could however be observed, and these are explored through cluster analysis of total fixation duration, see Sec. 3.3. Finally, the effect of auditory feedback on individual gaze behaviors are dis-

[6] Sound examples can be found at: `https://kth.box.com/s/2bhx2n2wrvq3na582h2qasfpqrxu0zod`

[7] Analysis was done separately for respective condition.

[8] Continuous data was analyzed using a Friedman rank test for repeated measures and count data was analyzed using a mixed effect regression model with a Poisson error distribution.

cussed and explored through sonifications of gaze trajectories, see Sec. 3.4.

3. RESULTS

In order to test the hypotheses, an index of the eye tracking metric (fixation duration or fixation count) for the ball AOI divided by the metric for both the goal AOI and the ball AOI was defined, according to Eq. (1):

$$I = \frac{FixBall}{FixGoal + FixBall} \tag{1}$$

This index simplifies the interpretation of whether the participant was focusing mostly on the ball or the goal AOI without having to take data from both AOIs into consideration simultaneously.

3.1 Total Fixation Duration

Total fixation duration measures the total duration of all fixations within a defined area of interest. In the current study this measure represents the total time that a participant fixated on the goal, ball or the entire screen, respectively. We calculated total fixation duration on the entire screen for the first 15 throwing gestures. Descriptive statistics for this measure are presented in Tab. 1. We can observe that the highest median value was found for condition VH. Lowest median value was observed for condition AV2.

Filtering recordings to include gaze data only from when the participants were grasping the virtual ball, the total fixation duration index (according to Eq. 1) was computed for each participant and condition. As opposed to the measure presented in Tab. 1, this measure specifically evaluates the effect of movement sonification (not the effect of other auditory events caused by e.g. scoring a goal) on eye tracking metrics, since such data has been filtered out. A total fixation duration index higher than 0.5 indicates longer fixation on the ball AOI than on the goal AOI (when grasping the ball). Descriptive statistics for the measure is presented in Tab. 2. Highest median value was found for the haptic condition AVH2. Lowest median value was observed for the nonhaptic condition AV2. Total fixation duration index per participant and condition is visualized in Fig. 4.

Condition	Median	IQR
V	137.39	21.72
AV1	137.96	24.06
AV2	125.92	25.44
VH	142.73	44.49
AVH1	132.39	19.39
AVH2	131.92	31.97

Table 1. Total fixation duration on screen (seconds) for full duration of first 15 attempts to throw the ball into the goal. IQR=inter quartile range. Red cells : highest median value. Green cell: lowest median value.

Condition	Median	IQR
V	0.53	0.77
AV1	0.43	0.61
AV2	0.38	0.70
VH	0.49	0.69
AVH1	0.46	0.60
AVH2	0.66	0.70

Table 2. Total fixation duration index for first 15 throwing gestures, filtered to include data only when grasping the ball. IQR=inter quartile range. Red cell: highest median value. Green cell: lowest median value.

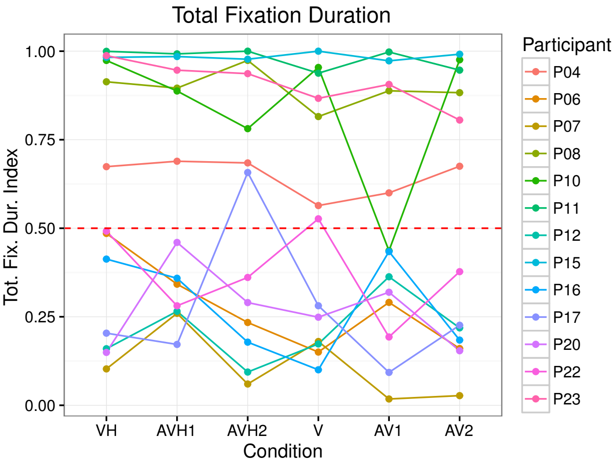


Figure 4. Total fixation duration index values for first 15 throwing gestures, filtered to include data only when grasping the ball. Participants above the dashed red line focused longer time on the ball than on the goal AOI.

3.2 Total Fixation Count

Total fixation count measures the total number of fixations within an area of interest. We calculated total fixation count on the entire screen for the first 15 throwing gestures. In the descriptive statistics table Tab. 3, we can observe that the highest median value was found for condition V. Lowest median value was observed for condition AV1.

Filtering data to include gaze data only from when the ball was grasped, total fixation count index (as described in Eq. 1) was computed for each participant and condition. A total fixation count index higher than 0.5 indicates more fixations on the ball AOI than on the goal AOI (when grasping the ball). Descriptive statistics for the measure is presented in Tab. 4. Highest median value was found for the haptic condition AVH2. Lowest medians were observed for the nonhaptic conditions AV1 and AV2.

3.3 Cluster Analysis - Total Fixation Duration

In order to explore similar behavioral patterns among participants, a K-means clustering analysis was performed. Appropriate number of clusters was decided after visually inspecting patterns in Fig. 4. The number of clusters was set to 5, and the initial cluster centers were evaluated based on the data, specifying random starting assignments to 20 and selecting the solution with the lowest within cluster

Condition	Median	IQR
V	211	88
AV1	183	80
AV2	200	74
VH	199	127
AVH1	188	102
AVH2	196	74

Table 3. Total fixation count on screen for full duration of first 15 attempts to throw the ball into the goal. IQR=inter quartile range. Red cell : highest median value. Green cell: lowest median value.

Condition	Median	IQR
V	0.59	0.31
AV1	0.53	0.23
AV2	0.53	0.31
VH	0.64	0.55
AVH1	0.64	0.33
AVH2	0.65	0.40

Table 4. Total fixation count index for first 15 throwing gestures. IQR=inter quartile range. Red cell: highest median value. Green cell: lowest median value.

variation. The total variance described by clusters was found to be 94.2 %. The cluster centers are given in Tab. 5. We could observe differences between conditions in the different cluster groups, but the effect of sound model was not consistent for all clusters. There were three clusters where several participants showed somewhat similar behaviors (2,3,5) and two single-participant clusters (1,3).

Overall, participants in cluster 3 had longer fixation durations on the goal AOI than on the ball AOI. Participants in cluster 5 showed the opposite behavior. As seen in Fig. 5, we can observe that 7 participants (belonging to clusters 2 and 3) generally had longer fixation durations on the goal AOI than on the ball AOI. A total of 6 participants (cluster 1,4 and 5) had longer fixation durations on the ball AOI.

Overall performance, measured as the sum of the error rates[9] for all conditions, was computed for each participant. In order to explore if there was a difference in performance between clusters, mean overall performance per cluster was computed. Mean error rate was 35.0 errors for cluster 1, 37.5 (median 37.5) for cluster 2, 32.0 (median 26.0) for cluster 3, 52.0 for cluster 4 and 95.0 (median 96.0) for cluster 5.

Cluster	VH	AVH1	AVH2	V	AV1	AV2
1	0.97	0.89	0.78	0.95	0.44	0.96
2	0.35	0.23	0.51	0.40	0.14	0.30
3	0.26	0.34	0.17	0.17	0.28	0.15
4	0.67	0.68	0.68	0.56	0.60	0.68
5	0.97	0.95	0.97	0.90	0.94	0.91

Table 5. Cluster centers per condition.

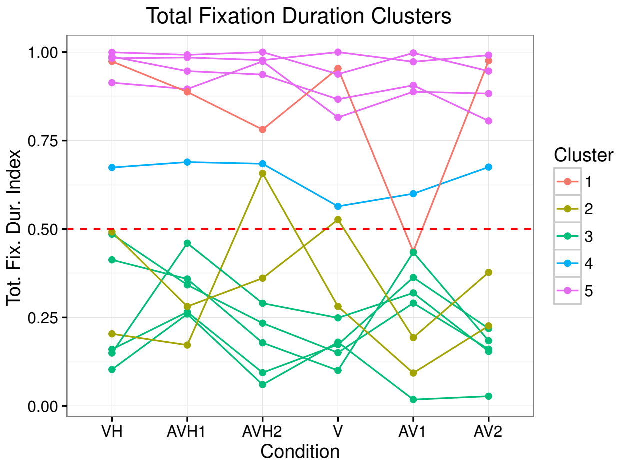


Figure 5. Cluster membership.

3.4 Individual Behaviors

Since very heterogeneous behaviors were observed among the participants we found it relevant to assess the effect of feedback condition for each participant individually. In this section, the gaze behavior of one participant from each cluster is discussed in detail. The metrics that are discussed in the following section come from the segmented data and only takes figures from when the ball was grasped into account. Total fixation duration- and total fixation count data for these individuals is presented in Tab. 6. Sonifications of these participants' behaviors when grasping the ball are also presented [10] in an attempt to explore temporal aspects and patterns in the gaze data. In these simple sonifications (based on sine waves with an applied reverb), the vertical coordinate of the averaged left- and right eye gaze points on the screen was mapped to pitch (increasing pitch for increasing vertical position) and the averaged horizontal coordinate was mapped to panning. Each throwing gesture was separated by a one second break. Interestingly, different behavioral patterns in cluster 3 and 5 are easily identified, for example by listening to the sonifications produced by gaze trajectories of participant 12 and 15.

3.4.1 *Cluster 1 - Participant 10*

This participant was classified separately in a solo cluster. The total error rate for all conditions for this participant was 35.0 errors. The participant performed the task in the following condition order: AV1, AVH2, AV2,

	VH	AVH1	AVH2	V	AV1	AV2
P04	21.59/28	31.4/48	26.77/45	15.98/30	18.02/30	19.82/29
P10	14.51/25	19.43/25	9.96/20	19.72/29	7.24/20	14.04/18
P12	3.05/13	5.04/28	2.10/14	3.13/10	7.44/28	5.43/19
P15	45.99/40	39.00/39	35.72/41	46.83/29	43.16/36	34.57/28
P22	16.26/57	5.81/25	6.65/31	29.67/77	4.29/22	12.25/35

Table 6. Total fixation duration (red) and total fixation count (blue) on ball AOI for participant P04, P10, P12, P15 and P22.

[9] For each condition, error rate was defined as the number of unsuccessful attempts to score a goal for a total of 15 achieved hits.

[10] Recordings of the sonified data are available at `https://kth.box.com/s/qgknhxjz7ul4sf3fi0oyqt7aipmun3v2`.

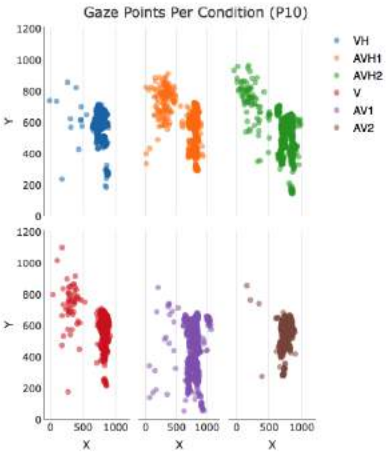


Figure 6. Gaze behavior of participant 10. Points correspond to the coordinates of the averaged left- and right eye gaze position on the screen.

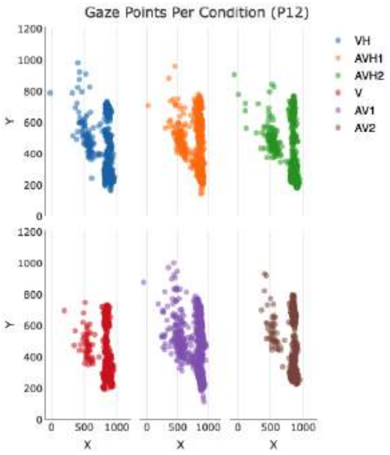


Figure 7. Gaze behavior of participant 12. Points correspond to the coordinates of the averaged left- and right eye gaze position on the screen.

AVH1, V and VH. The participant focused mostly on the ball, in most conditions (all apart from AV1). The coordinates of the averaged left- and right eye gaze positions are displayed in Fig. 6. The plot suggests that there might be a difference between conditions (the points are more or less scattered in different conditions). For the ball AOI, this participant had total fixation durations in the range between 7.24 and 19.72 seconds (lowest for condition AV1 and highest for condition V) and total fixation counts in the range between 18 and 29 counts (lowest for AV2 and highest for condition V).

3.4.2 Cluster 2 - Participant 22

Generally, participants belonging to cluster 2 focused longer on the ball AOI for the swishing sound model than for the creaking sound model. The total error rate for all conditions for participants in this cluster was 37.5 (median 37.5). Participant 22 performed the experiment in the following condition order: V, AV2, VH, AV1, AVH1 and AVH2. For most of the conditions, this particular participant focused longer time on the goal AOI than on the ball AOI. For the ball AOI, participant 22 had total fixation durations in the range between 4.29 and 29.67 seconds (lowest for AV1 and highest for V) and total fixation counts in the range between 22 and 77 counts (lowest for AV1 and highest for condition V). By listening to the sonified gaze trajectories, one can observe that condition V stands out from the other conditions; this condition had overall longer gaze durations, compared to other nonhaptic conditions. Interestingly, we can observe from the metrics in Tab. 6 that this participant focused more on the ball for conditions where no sound was present.

3.4.3 Cluster 3 - Participant 12

Participants belonging to cluster 3 generally focused longer on the goal AOI than on the ball AOI. The total error rate for all conditions for participants in this cluster was 32 (median 26), i.e. the best performance of all clusters. Contrary to the behavior of participants in cluster 2, most participants in cluster 3 focused longer on the ball AOI for the creaking sound model than for the swishing sound model. For participant 12, this tendency can be observed both for haptic and nonhaptic conditions. The participant performed the experiment in the following condition order: AVH1, AV1, AVH2, AV2, V and VH. The coordinates of the averaged left- and right eye gaze positions are displayed in Fig. 7. As seen in the figure, the gaze points are distributed in a similar pattern in all conditions. For the ball AOI, the participant had total fixation durations in the range between 3.05 and 7.44 seconds (lowest for VH and highest for AV1) and total fixation counts in the range between 10 and 28 counts (lowest for V and highest for AV1). As indicated by the plots presented in Fig. 7, the sonifications of this participant's gaze behavior do sound different in different conditions.

3.4.4 Cluster 4 - Participant 04

This participant was classified separately in a solo cluster. The total error rate for all conditions for this participant was 52.0. The participant performed the experiment in the following condition order: AVH1, VH, V, AVH1, AV2 and AV1. As indicated by the metrics in Tab. 5, this participant focused approximately as much on the goal AOI as on the ball AOI in all conditions. For the ball AOI, participant 4 had total fixation durations in the range between 15.98 and 31.40 seconds (lowest for V and highest for AVH1) and total fixation counts in the range between 28 and 48 counts (lowest for VH and highest for AVH1). The V condition stands out compared to other conditions.

3.4.5 Cluster 5 - Participant 15

Participants in cluster 5 showed the opposite behavior of participants in cluster 3: they generally focused more on the ball AOI than on the goal AOI. The total error rate for all conditions for participants in this cluster was 95.0 (median 96.0), i.e. the highest error rate of all clusters. Participant 15 performed the conditions in the following order: V, AV1, VH, AVH1, AV2 and AVH2. The coordinates of the averaged left- and right eye gaze positions are displayed in Fig. 8. As seen in the figure, gaze trajectories appear to

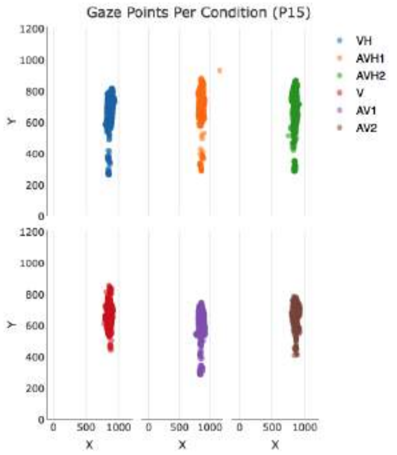


Figure 8. Gaze behavior of participant 15. Points correspond to the coordinates of the averaged left- and right eye gaze position on the screen.

have been very focused for this participant. Sonifications reveal that the participant showed similar gaze trajectory patterns for all conditions. This is also clearly shown in Fig. 4. In general, the participant had rather long total fixation durations, in the range between 34.57 and 46.83 seconds (lowest for AV2 and highest for V) and total fixation counts in the range between 28 and 40 counts (lowest for AV2 and highest for AVH1), for the ball AOI.

4. DISCUSSION

In this paper, we explored the effect of auditory feedback on gaze behavior in haptic versus nonhaptic conditions when manipulating an object in a virtual environment. Since there can be substantial differences in gaze behavior between participants even for identical tasks, it is good practice to use a within-participant design in these contexts, in order to make valid comparisons of performance [20]. As in [11], we observed very heterogeneous behaviors among participants. The effect of movement sonification was thus evaluated on an individual level, similarly to previous studies on the effect of sonification on gaze behavior [10, 11]. Possibly due to the limited sample size of this study (and the fact that a substantial part of the eye tracking data had to be discarded since the percentage of usable gaze data was too low), no significant differences between feedback conditions could be observed. We can therefore not make any general assumptions about the hypotheses defined in section 2.1. Some general trends could, however, be identified.

For the full duration of the first 15 attempts to throw the ball into the goal, the total fixation count median value was highest for condition V, in which no auditory or haptic feedback was provided. For total fixation duration, the median was highest for condition VH (followed by condition V). For fixation count, the V condition also had higher median value than the AV1 and AV2 conditions, and the VH condition had higher median value than the AVH1 and AVH2 conditions. Although no significant difference between conditions could be observed, these results indicate that it would be worth investigating if the presence of auditory feedback affects gaze behavior in future studies.

Cluster analysis revealed that the participants could be divided into five different clusters in which similar gaze behaviors were found. Within only 13 participants, 5 different sorts of behavior were observed. In total, 6 participants had longer fixation duration on the ball AOI than the goal AOI. A total of 7 participants showed the opposite behavior. On an individual level, we could observe that some participants appeared to have been affected by the presented auditory feedback (for example, differences appeared for participants in cluster 2 and 3), whereas others showed very similar gaze behavior across all conditions. Even when an effect of auditory models seemed to have been present, the behaviors were not consistent. For example, participants in cluster 2 showed the opposite behavior of participants in cluster 3.

The performed interviews shed some light on the effects of auditory feedback. A few participants stated that it was easier to aim in the conditions with auditory feedback. A possible explanation, given by one of the participants, was: "*It was easier to get an understanding of where the ball was located [in auditory conditions]. I could make use of how much sound that I needed [...] You could find a rhythm in the sound that you could follow.*". This quote indicates that the sonification of the movement gave some information that otherwise had to be provided graphically. Another participant further emphasized this by stating: "*I liked the sound because I could hear which velocity I had. Then I could also focus more on the goal and the direction of the ball, instead of just on the ball.*".

Mean performance varied between the clusters. Sorting cluster by best to worst mean performance, we obtain: cluster 3, cluster 1, cluster 2, cluster 4 and cluster 5. We can observe that best performance was observed for participants belonging to clusters where the longer fixation duration was on the goal AOI, rather than on the ball AOI. The cluster with participants who focused longer on the ball AOI than on the goal AOI (cluster 5) had worst overall performance.

Large differences in eye movements between participants encourage large datasets. To address the issue of heterogeneous behaviors, we propose a future more controlled experiment with larger sample size to evaluate the effect of sonification on gaze behavior. The current study focused specifically on movement sonification (based on position and velocity). Possibly, auditory feedback might have been more powerful in terms of affecting gaze behavior if the sonification had focused on guiding the user in terms of signaling if the performed movement was "correct" or not (i.e. error sonification).

Finally, we encourage the use of sonification for rapid exploration of large sets of gaze data. Sonifying gaze trajectories enables the listener to identify behaviors not easily discovered in 2D plots. For example, patterns such as rhythm and overall duration of a gaze trajectory are easily identified using sonification. In this particular study, sonification enabled us to confirm different gaze behaviors identified in the cluster analysis, such as the distinction between focusing mostly on the ball versus goal AOI for cluster 3 and 5.

5. CONCLUSIONS

This exploratory study focused on the effect of auditory feedback on gaze behavior. Analysis of eye tracking metrics revealed several clusters of different gaze behaviors among only 13 participants. Some of the participants appeared to have been affected by the presence of auditory feedback, whereas other showed similar gaze trajectory patterns across conditions. More thorough studies are required in order to be able to draw any general conclusions regarding the effect of auditory feedback in this context. This exploratory study raises intriguing questions that motivate future large-scale studies on the effect of auditory feedback on gaze behavior.

Acknowledgments

This work was supported by the Swedish Research Council (Grant No. D0511301). It was also partially funded by a grant to Roberto Bresin from KTH Royal Institute of Technology.

6. REFERENCES

[1] J. Vroomen and B. d. Gelder, "Sound enhances visual perception: cross-modal effects of auditory organization on vision." *Journal of experimental psychology: Human perception and performance*, vol. 26, no. 5, p. 1583, 2000.

[2] L. Shams, Y. Kamitani, and S. Shimojo, "Illusions: What you see is what you hear," *Nature*, vol. 408, no. 6814, pp. 788–788, 2000.

[3] J. Mishra, A. Martínez, and S. A. Hillyard, "Effect of attention on early cortical processes associated with the sound-induced extra flash illusion," *Journal of cognitive neuroscience*, vol. 22, no. 8, pp. 1714–1729, 2010.

[4] S. L. Calvert and T. L. Gersh, "The selective use of sound effects and visual inserts for children's television story comprehension," *Journal of Applied Developmental Psychology*, vol. 8, no. 4, pp. 363–375, 1987.

[5] A. R. Seitz, R. Kim, and L. Shams, "Sound facilitates visual learning," *Current Biology*, vol. 16, no. 14, pp. 1422–1427, 2006.

[6] G. Song, D. Pellerin, and L. Granjon, "Different types of sounds influence gaze differently in videos," *Journal of Eye Movement Research*, vol. 6, no. 4, pp. 1–13, 2013.

[7] G. Song, D. Pellerin, and L. Granjon, "Sound effect on visual gaze when looking at videos," in *Signal Processing Conference, 2011 19th European*. IEEE, 2011, pp. 2034–2038.

[8] A. Coutrot, N. Guyader, G. Ionescu, and A. Caplier, "Influence of soundtrack on eye movements during video exploration," *Journal of Eye Movement Research*, vol. 5, no. 4, p. 2, 2012.

[9] X. Min, G. Zhai, Z. Gao, C. Hu, and X. Yang, "Sound influences visual attention discriminately in videos," in *Quality of Multimedia Experience (QoMEX), 2014 Sixth International Workshop on*. IEEE, 2014, pp. 153–158.

[10] V. Losing, L. Rottkamp, M. Zeunert, and T. Pfeiffer, "Guiding visual search tasks using gaze-contingent auditory feedback," in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*. ACM, 2014, pp. 1093–1102.

[11] E. Boyer, "Continuous auditory feedback for sensorimotor learning," Ph.D. dissertation, Université Pierre et Marie Curie-Paris VI, 2015.

[12] A. Poole and L. J. Ball, "Eye tracking in hci and usability research," *Encyclopedia of human computer interaction*, vol. 1, pp. 211–219.

[13] "Computer interface evaluation using eye movements: methods and constructs," *International Journal of Industrial Ergonomics*, vol. 24, no. 6, pp. 631 – 645, 1999.

[14] A. Duchowski, *Eye tracking methodology: Theory and practice*. Springer Science & Business Media, 2007, vol. 373.

[15] M. A. Just and P. A. Carpenter, "Eye fixations and cognitive processes," *Cognitive Psychology*, vol. 8, no. 4, pp. 441 – 480, 1976.

[16] E. Frid, J. Moll, R. Bresin, and E.-L. Sallnäs-Pysander, "Influence of movement sonification on performance in a virtual throwing task performed with and without haptic feedback," *Manuscript submitted for publication.*

[17] F. Conti, F. Barbagli, D. Morris, and C. Sewell, "{CHAI} an open-source library for the rapid development of haptic scenes," in *Proceedings of the IEEE World Haptics Conference*, 2005. [Online]. Available: http://www.chai3d.org/

[18] M. Wright, "Open sound control: an enabling technology for musical networking," *Organised Sound*, vol. 10, no. 03, pp. 193–200, 2005.

[19] S. D. Monache, P. Polotti, and D. Rocchesso, "A toolkit for explorations in sonic interaction design," in *Proceedings of the 5th Audio Mostly Conference: A Conference on Interaction with Sound*. ACM, 2010, p. 1.

[20] J. H. Goldberg and A. M. Wichansky, "Chapter 23 - eye tracking in usability evaluation: A practitioner's guide," in *The Mind's Eye*, J. Hyn, R. Radach, and H. Deubel, Eds. Amsterdam: North-Holland, 2003, pp. 493 – 516.

Auralization in space and in rooms of arbitrary D dimensions

Felipe Orduña-Bustamante
Centro de Ciencias Aplicadas y Desarrollo Tecnológico,
Universidad Nacional Autónoma de México,
Circuito Exterior S/N, Ciudad Universitaria,
Del. Coyoacán, C.P. 04510, México D.F., MÉXICO.
felipe.orduna@ccadet.unam.mx

ABSTRACT

Signal processing and computing procedures are presented to imprint, into any given audio signal (preferably obtained from a dry anechoic recording), the temporal and spectral characteristics of radiation of simple sources in free field, and in reverberant rectangular rooms, that could virtually be obtained in space of any spatial dimensions $D{\geq}1$. These techniques are based on mathematically exact solutions of the conventional acoustic field equations, usually expressed and solved for space of three dimensions or less, but generalized formally to describe the analogue to sound waves in space of higher dimensions. Processing as presented here is monaural, but the mathematical tools exist to imprint a hyper-angular dependence in higher dimensions, and to map that into conventional binaural presentations. Working computer implementations, and sound samples are presented. The foreseen applications include: sonic art, virtual reality, extended reality, videogames, computer games, and others.

1. INTRODUCTION

The subject of high-dimensional wave equations has been extensively developed in Applied Quantum Mechanics, in dealing with quantum systems with many bodies [1–4]. Schrodinger's equation for a system of N bodies, involves a wave equation in a coordinate or momentum space of dimension $D = 3N - 1$, after removing the motion of the center of mass. Most of that analysis separates a radial part of the solution, from a hyper-angular part [5–7]. This article deals with the adaptation of radial part solutions, to solve the corresponding equations of acoustics for a radial sound pressure field in space of arbitrary D dimensions. Radiation of simple acoustic sources are then combined with the source image method [8], in order to simulate a reverberant sound field in a room of D dimensions. Further work on this subject will probably involve dealing with the hyper-angular part of the higher-dimensional wave equation [9, 10], and devising some sort of mapping the resulting audio into conventional binaural sound presentations. Formulations using finite-difference time-domain FDTD methods have also been developed by others [11, 12].

2. RADIAL SOUND WAVES IN D DIMENSIONS

The acoustic field equations for the sound pressure $p(\mathbf{x}, \omega)$, and particle fluid velocity $u(\mathbf{x}, \omega)$ in the frequency domain ω, assuming a complex harmonic time dependence of the form $e^{+j\omega t}$, are as follows:

$$\operatorname{grad} p(\mathbf{x}, \omega) = -jkZ_0 \mathbf{u}(\mathbf{x}, \omega), \tag{1}$$
$$\operatorname{div} \mathbf{u}(\mathbf{x}, \omega) = -jkZ_0^{-1} p(\mathbf{x}, \omega); \tag{2}$$

where $k = \omega/c_0$ is the acoustic wavenumber, $Z_0 = \rho_0 c_0$ is the characteristic acoustic impedance, c_0 is the speed of sound, and ρ_0 is mass density of the propagation medium. These can be written for a radial field in D dimensions in the form:

$$\frac{\partial p}{\partial r} = -jkZ_0 u_r(r), \tag{3}$$
$$\frac{1}{r^{D-1}} \frac{\partial}{\partial r}\left(r^{D-1} u_r\right) = -jkZ_0^{-1} p(r). \tag{4}$$

These remain similar to its usual form, as in cylindrical, or spherical coordinates in $D = 2,\ 3$.

The radial part of the in-homogeneous Helmholtz equation can be written as follows:

$$\nabla^2 p(r) + k^2 p(r) = -q(r), \tag{5}$$
$$\frac{\partial^2 p}{\partial r^2} + (D-1)\frac{1}{r}\frac{\partial p}{\partial r} + k^2 p(r) = -q(r), \tag{6}$$
$$\frac{1}{r^{D-1}} \frac{\partial}{\partial r}\left(r^{D-1}\frac{\partial p}{\partial r}\right) + k^2 p(r) = -q(r). \tag{7}$$

The substitution $p(r) = \phi(kr)/(kr)^{D/2-1}$, turns this into Bessel's equation of order $\nu = D/2 - 1$, the order ν being zero, or a positive integer, when the space dimension D is even, and half-integer when D is odd. Solutions for $\phi(kr)$, correspond to Hankel functions of order ν, in terms of which, outgoing radial waves for the sound pressure, when the source term is a spatial Dirac point distribution $q(r) = \delta(r)$, can be written as follows:

$$p(r, \omega) = \frac{\mathcal{A}\, H_\nu^{(2)}(kr)}{(kr)^{D/2-1}}, \tag{8}$$
$$\approx \frac{\sqrt{2/\pi}\mathcal{A}\, e^{-j[kr-(D-1)\pi/4]}}{(kr)^{(D-1)/2}},\ kr \gg 1; \tag{9}$$

where $\mathcal{A}$ is the complex amplitude of the sound pressure, with corresponding expressions for the particle velocity. Additional terms can also include incoming waves.

2.1 The simple monopole source in D dimensions

The radiation transfer function $\mathcal{M}_D(r,\omega)$ of a simple monopole source can be defined as the ratio between the outgoing sound pressure wave amplitude in the far field, to the volume velocity in the near field, $Q_0(\omega)$, with $q(r) = j\omega\rho_0\,\delta(r)$, which in D dimensions works out as follows:

$$\mathcal{M}_D(r,\omega) \;=\; (Z_0/2)\left(\frac{jk}{2\pi r}\right)^{(D-1)/2} e^{-jkr}. \quad (10)$$

Calculation of the inverse Fourier transform of $p(r,\omega) = \mathcal{M}_D(\omega)Q_0(\omega)$, involves considering the time delay operator: $e^{-j\omega r/c_0} \longleftrightarrow \delta(t - r/c_0)$, the derivative operator: $(j\omega)^\nu \longleftrightarrow d^\nu/dt^\nu$, and for D even, the half-integration operator: $(j\omega)^{-1/2} \longleftrightarrow (\pi t)^{-1/2}$, implied by the required half-integer (fractional) derivatives. The corresponding expressions in the time domain, are as follows:

$$\begin{aligned} p(r,t) &= \frac{(Z_0/2)}{(1/2\pi r c_0)^{(D-1)/2}} \times && (11)\\ &\quad Q_0^{[(D-1)/2]}(t - r/c_0),\ D\,\text{odd}, \\ &= \frac{(Z_0/2)}{(1/2\pi r c_0)^{(D-1)/2}} \times && (12)\\ &\quad \int_0^\infty \frac{Q_0^{[D/2]}(t - r/c_0 - \tau)}{(\pi\tau)^{1/2}}\,d\tau,\ D\,\text{even}; \end{aligned}$$

where Fourier transform pairs are indicated by their arguments: $p(r,t) \longleftrightarrow p(r,\omega)$, $Q_0(r,t) \longleftrightarrow Q_0(r,\omega)$, and with time derivatives indicated by $Q_0^{[\nu]}(t) = d^\nu Q_0(t)/dt^\nu$. The first expression, Eq. (11), is formally valid also for D even, but in that case, a fractional derivative is implied, which should be understood as in the second expression, Eq. (12). Figure 1 shows sound pressure pulses radiated identically and simultaneously by two monopole sources at duplicate distances in space of dimensions $D = 1$ to 6.

3. RECTANGULAR ROOMS IN D DIMENSIONS

3.1 The image method in one dimension, $D = 1$

In a one dimensional cavity with rigid ends, the characteristic functions $\phi_m(x)$, and characteristic values k_m, satisfying the Helmholtz equation, and the rigid boundary conditions, are as follows:

$$\begin{aligned} \frac{\partial^2\phi_m}{\partial x^2}\phi_m + k_m^2\phi_m(x) &= 0, && (13)\\ \partial\phi_m/\partial x &= 0,\ x = 0, L, && (14)\\ \phi_m(x) &= (2/L)^{1/2}\cos k_m x, && (15)\\ k_m &= m\pi/L, && (16)\\ \int_0^L dx\,\phi_m(x)\phi_n(x) &= \delta_{mn}. && (17) \end{aligned}$$

The fundamental solution, or Green's function, $G(x,y)$, satisfies the in-homogeneous Helmholtz equation with a concentrated point source:

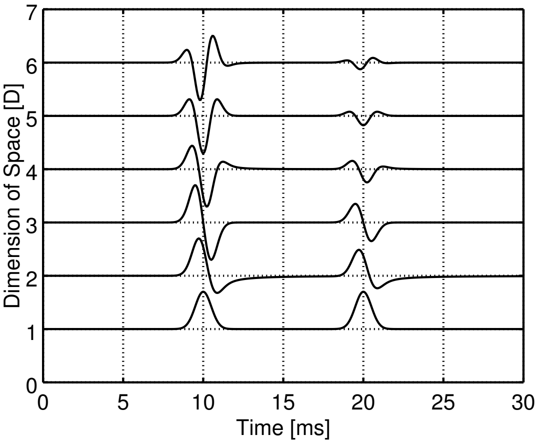


Figure 1. Sound pressure pulses radiated identically and simultaneously by two monopole sources at duplicate distances in space of dimensions $D = 1$ to 6. Absolute peak amplitudes are normalized to 0.7 vertical units.

$$\begin{aligned} \frac{\partial^2 G}{\partial x^2} + k^2 G(x,y) &= -\delta(x-y), && (18)\\ \frac{\partial G}{\partial x} &= 0,\ x = 0, L. && (19) \end{aligned}$$

The fundamental solution can be expressed in terms of the characteristics functions, as follows:

$$\begin{aligned} G(x,y) &= \sum_m \frac{\phi_m(x)\,\phi_m(y)}{k_m^2 - k^2}, && (20)\\ G(x,y) &= \left(\frac{2}{L}\right)\sum_m \frac{\cos k_m x\,\cos k_m y}{k_m^2 - k^2}. && (21) \end{aligned}$$

This representation can be transformed into the following, in terms of image sources driven at unit volume velocity, $p(x,y) = jkZ_0\,G(x,y)$, or unit particle velocity in $D = 1$:

$$\begin{aligned} p(x,y) &= \left(\frac{Z_0}{2}\right)\sum_{n=-\infty}^{+\infty} e^{-jk|x-y_n^+|} + && (22)\\ &\qquad e^{-jk|x-y_n^-|}, \\ &= \sum_{n=-\infty}^{+\infty} \mathcal{M}_{D=1}(|x - y_n^+|,\omega) + && (23)\\ &\qquad \mathcal{M}_{D=1}(|x - y_n^-|,\omega). \end{aligned}$$

This is the superposition of simple sound sources (monopoles) in two different regularly spaced, infinite arrays of image source locations: $y_n^+ = 2\,L\,n + y$, $y_n^- = 2\,L\,n - y$, for all possible integer values of n. In order to take into account sound absorption at the walls, additional factors can also be included in integer powers of the corresponding reflection coefficients, depending on the number of wall reflections that each image source implies.

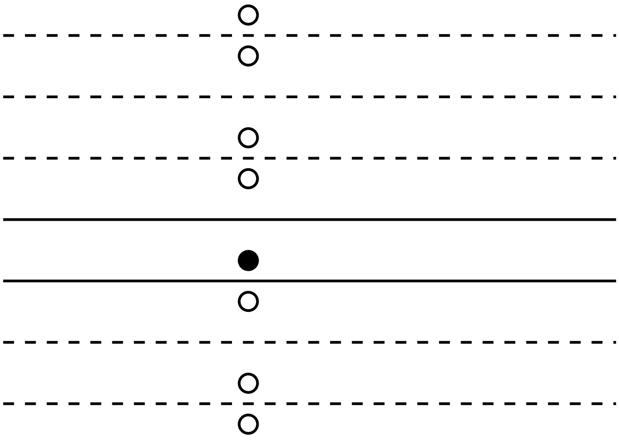


Figure 2. Distribution of image sources in $D = 1$.

3.2 The image method in D dimensions

The image method can be extended in a very straightforward fashion to arbitrary dimension D, as follows:

$$
\begin{aligned}
p(\mathbf{x},\mathbf{y}) &= \sum_{\mathbf{n}=-\infty}^{+\infty} \mathcal{M}_{D_s}(|\mathbf{x}-\mathbf{y_n}^{+}|,\omega) + \qquad (24)\\
&\qquad \mathcal{M}_{D_s}(|\mathbf{x}-\mathbf{y_n}^{-}|,\omega),\\
\mathbf{x} &= (x_1, x_2, x_3, ..., x_D), \qquad (25)\\
\mathbf{y} &= (y_1, y_2, y_3, ..., y_D), \qquad (26)\\
\mathbf{n} &= (n_1, n_2, n_3, ..., n_D), \qquad (27)\\
\mathbf{y_n}^{+} &= 2\,\mathbf{L_n} + \mathbf{y}, \qquad (28)\\
\mathbf{y_n}^{-} &= 2\,\mathbf{L_n} - \mathbf{y}, \qquad (29)\\
\mathbf{L_n} &= (n_1L_1, n_2L_2, n_3L_3, ..., n_DL_D), \qquad (30)
\end{aligned}
$$

where the sum (now a D-fold sum) runs over all possible combinations of the index vector $\mathbf{n}$. Note that a generalization has been included, in that the space dimension D_s of the monopole transfer function $\mathcal{M}_{D_s}(r,\omega)$, is not necessarily the same as the space dimension of the room D. The formulation allows any kind of relationship between D_s, and D; D_s can be smaller, greater, or equal to D. In our physical space, and for radial sound fields with no angular dependence, these are restricted to $(D, D_s) \leq 3$, but still, both can be related in any way within this restriction. Figure 2 shows the distribution of image sources in $D = 1$. Figure 3 shows the distribution of image sources in $D = 2$. In these illustrated cases, with the listener in line, or in plane with the image sources, the room geometry has dimensions $D = 1, 2$, as implied by the distribution of image sources. However, the source transfer functions adopt the dimension of the embedding space: $D_s = 3$.

4. IMPLEMENTATION AND RESULTS

Implementation involves the following procedures:

- Define the room geometry (dimension D, etc.), wall reflection coefficients, sound source location, and listening location. Optionally, a different dimension D_s for the source transfer function.

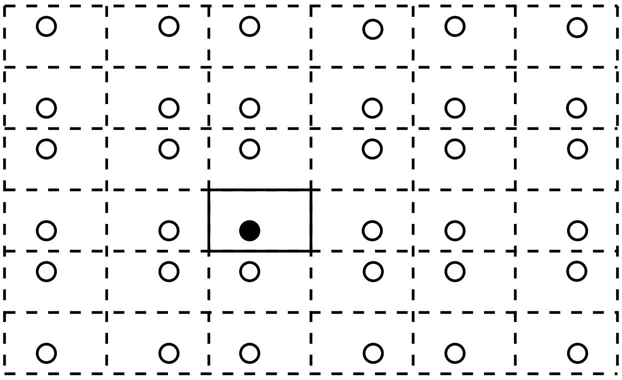


Figure 3. Distribution of image sources in $D = 2$.

- Generate the impulse response using the method of image sources, Eq. 24, up to a certain order.

- For each image source, take into account the time delay: r/c_0, attenuation with distance: $1/r^{(D_s-1)/2}$, and wall absorption, but do not implement at this stage the frequency weighting implied by the time derivatives. See Eqs. 10, 11, 12, Section 2.1.

- Take an audio signal, preferably a dry anechoic recording, and implement a fast convolution with the room's impulse response. This time, also apply the frequency weighting: $(j\omega)^{(D_s-1)/2}$.

Deferring the frequency weighting to a later stage in the processing, allows the option of removing, or changing it, preserving the image source calculation.

Impulses responses were generated coding in C and GNU Octave, using the parameters of one of the rooms simulated in $D = 3$, with $f_s = 8$ kHz, and $37,500$ image sources (up to about order $n = \pm 16$), by Allen and Berkley [8]:

$$
\begin{aligned}
\mathbf{L} &= (3.43,\ 5.15,\ 4.29), \qquad (31)\\
\mathbf{x} &= (2.14,\ 0.43,\ 2.57), \qquad (32)\\
\mathbf{y} &= (1.29,\ 4.29,\ 1.72), \qquad (33)\\
\mathbf{R}_0 &= (0.70,\ 0.90,\ 0.90), \qquad (34)\\
\mathbf{R}_L &= (0.70,\ 0.90,\ 0.90), \qquad (35)
\end{aligned}
$$

with longitudinal units in meters (after the originals reported in sampling periods), and sets of reflection coefficients: $\mathbf{R}_0, \mathbf{R}_L$, whose components n are the reflection coefficients of the walls defined by $x_n = 0$, and $x_n = L_n$, respectively. Rooms of dimensions $D = 1$ to 6 were defined using these parameters, by including the first ($D = 1$), then successive ($D = 2$, 3), then recycling ($D = 4$, 5, 6) components from the data set. Sampling frequency was the same, $f_s = 8$ kHz, and image sources were calculated to order $n = \pm 64$, which for $D = 3$, add up to $N = 129^3 = 2,146,689$ image sources. Figure 4 shows impulse responses in rooms of space dimensions $D = 1$ to 6, $D_s = D$. Figure 5 shows impulse responses with monopole transfer functions of fixed dimension: $D_s = 3$. Both figures include frequency weightings: $(j\omega)^{(D_s-1)/2}$. Figure 6 shows the calculation times.

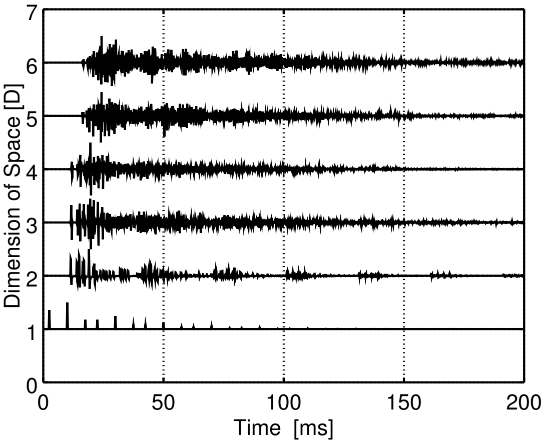


Figure 4. Impulse responses in rooms of space dimensions $D = 1$ to 6, $D_s = D$, including: $(j\omega)^{(D_s-1)/2}$. Absolute peak amplitudes are normalized to 0.5 vertical units.

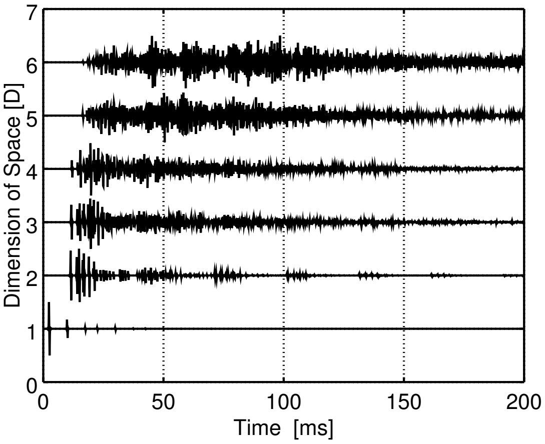


Figure 5. Impulse responses for source transfer functions of fixed dimension: $D_s = 3$, including: $(j\omega)^{(D_s-1)/2}$.

5. DISCUSSION

From a graphical analysis of the impulse responses, and also from the perceptual experience of listening to audio processed through them, it is possible to conclude that with increasing dimension D, there is increased and more intricate reverberation, with a consequently reduced intelligibility, and even with strong *aftermaths* at the higher dimensions. These are caused by the lumping together of reflections that get mapped into the same neighboring sampling periods (also perceptually) in certain portions of the impulse response. At the lower dimensions (especially $D = 1, 2$), there are more chances of *flutter* and *pulsation* effects, while at the higher dimensions, sound tends to increasingly acquire *hissing* and *chirping* effects.

When using a source transfer function of fixed dimension, Figure 5, reverberation effects are noticeably reduced for rooms with $D < D_s$, but are over-emphasized when $D > D_s$. This occurs because, compared to the normal radial decay $1/r^{(D-1)/2}$, the radial decay $1/r^{(D_s-1)/2}$ has little reach for $D < D_s$, but is more far reaching for $D > D_s$.

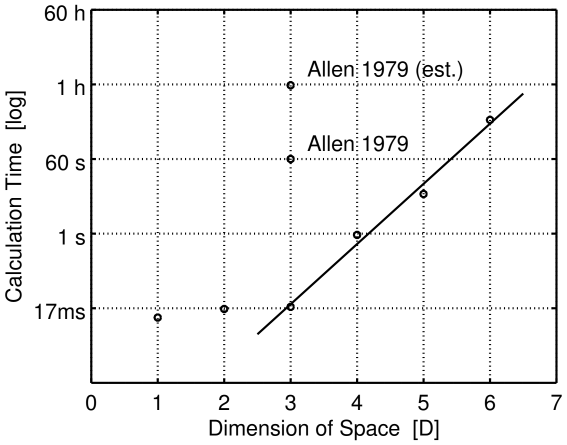


Figure 6. Calculation times for the generation of impulse responses in rooms of dimensions $D = 1$ to 6. Timings of Allen 1979 for $N = 37,500$ image sources [8], and also estimated (est.) for $N = 2,146,689$, used here for $D = 3$.

6. CONCLUSIONS

Auralization procedures have been presented to produce sound signals that correspond to the analogue of acoustic propagation in free field, and room reverberation, in spaces of arbitrary higher dimensions D. Processing as presented here is monaural, but the mathematical tools exist to imprint a hyper-angular dependence in higher dimensions, and to map that into conventional binaural presentations. Applications are foreseen in the fields of sonic art, virtual reality, extended reality, videogames, computer games, etc.

7. REFERENCES

[1] S.-H. Dong, *Wave Equations in Higher Dimensions*. Springer-Verlag, 2011.

[2] J. Avery, *Hyperspherial Harmonics: Applications in Quantum Theory*. Springer-Verlag, 1989.

[3] S. M. Al-Jaber, "Hydrogen Atom in N Dimensions," *International Journal of Theoretical Physics*, vol. 37, no. 4, pp. 1289–1298, 1998.

[4] H. Hosoya, "Back-of-Envelope Derivation of the Analytical Formulas of the Atomic Wave Functions of a D-Dimensional Atom," *International Journal of Quantum Chemistry*, vol. 64, pp. 35–42, 1997.

[5] G. Dassios and A. Fokas, "Methods for solving elliptic PDE's in spherical coordinates," *SIAM J. Appl. Math.*, vol. 68, no. 4, pp. 1080–1096, 2008.

[6] R. L. Liboff, "Complex Hyperspherical Equations, Nodal-Partitoning, and First-Excited-State Theorems in $\mathbb{R}^n$," *International Journal of Theoretical Physics*, vol. 41, no. 10, pp. 1957–1970, October 2001.

[7] W.-Q. Zhao, "Relation Between Dimensional and Angular Momentum for Radially Symmetric Potential in N-Dimensional Space," *Commun. Theor. Phys.*, vol. 46, no. 3, pp. 429–434, September 2006.

[8] J. B. Allen and D. A. Berkley, "Image method for efficiently simulating small-room acoustics," *Journal of the Acoustical Society of America*, vol. 65, no. 4, pp. 943–950, April 1979.

[9] J. S. Avery, "Harmonic polynomials, hypershperical harmonics, and atomic spectra," *Journal of Computational and Applied Mathematics*, vol. 233, pp. 1366–1379, 2010.

[10] T. Kereselidze, G. Ckdua, P. Defrance, and J. Ogilvie, "Derivations, properties and applications of Coulomb Sturmians defined in spheroidal coordinates," *Molecular Physics*, vol. 114, no. 1, pp. 148–161, 2016.

[11] Antti Kelloniemi and Vesa Välimäki and Patty Huang and Lauri Savioja, "Artificial reverberation using a hyper-dimensional FDTD mesh," in *13th European Signal Processing Conference*, IEEE Conference Publications, Ed., 2005, pp. 1–4.

[12] Antti Kelloniemi and Patty Huang and Vesa Välimäki and Lauri Savioja, "Hyper-Dimensional Digital Waveguide Mesh for Reverberation Modeling," in *CiteSeerX https://core.ac.uk/.* oai:CiteSeerX.psu:10.1.1.384.708, 2013-12-08, 2013.

AUDITORY, VISUAL AND SOMATOSENSORY LOCALIZATION OF PIANO TONES: A PRELIMINARY STUDY

Federico Fontana, Debora Scappin
Università di Udine
Dept. of Math., Comp. Sci. & Phys.
federico.fontana@uniud.it
scappin.debora@spes.uniud.it

Federico Avanzini
Università di Padova
Dept. of Information Engin.
avanzini@dei.unipd.it

Mattia Bernardi, Devid Bianco, Giorgio Klauer
Conservatorio “Cesare Pollini”
mattiabernardi9@gmail.com
biancodevid@gmail.com
giorgio.klauer@gmail.com

ABSTRACT

The paper presents an experiment in which subjects had to localize headphone-reproduced binaural piano tones while in front of a grand Disklavier instrument. Three experimental conditions were designed: when the fallboard was closed the localization was auditory only; when the fallboard was open the localization was auditory and visual, since the Disklavier’s actuated key could be seen moving down while the corresponding note was produced; when the listener actively played the note the localization was auditory, visual and somatosensory. In all conditions the tones were reproduced using binaural recordings previously acquired on the same instrument. Such tones were presented either transparently or by reversing the channels. Thirteen subjects participated in the experiment. Results suggest that if auditory localization associates the tone to the corresponding key, then the visual and somatosensory feedback refine the localization. Conversely, if auditory localization is confused then the visual and somatosensory channels cannot improve it. Further experimentation is needed to explain these results in relation with i) possible activation of the auditory precedence effect at least for some notes, and ii) potential locking of the sound source position that visual and/or somatosensory cues might cause when subjects observe a key moving down, or depress it.

1. INTRODUCTION

The industry of digital pianos has traditionally relied on the assumption that pianists are able to localize a tone while playing their instrument. The tone is often assumed to arrive at the listening point with an angle spanning the median plane and the key position of the corresponding note. This assumption has led to the design of several panning technologies: besides their use for improving a stereophonic loudspeaker reproduction of the recorded soundfield from an electronic instrument such as a digital piano [1], these technologies were felt especially necessary to render the same soundfield through headphones, or when the sound results from a synthesis model instead of a sample-based design [2].

This assumption finds no straightforward explanation in measurements of real piano soundfields. Contrarily to what one would initially expect, piano tones do not show radiation patterns that can be linked to the note position. These patterns, rather, depend on the modal characteristics of the soundboard, which absorbs almost all the energy that resonates within the strings and then radiates into the air according to its specific patterns of propagation [3]. Different soundboard regions radiate corresponding frequency components; the net result is that, once a tone is played, the pianist is enveloped by a soundfield having no relationship with the position in the keyboard of the corresponding note. This soundfield has been reproduced also in grand digital pianos mounting more expensive sound diffusion technologies [4].

Soundboard radiation prevails during the steady state, i.e., several tens of milliseconds after the hammer has excited the strings, which in their turn freely oscillate while gradually transferring their energy to the soundboard through the bridge. Conversely, before the hammer releases the strings the piano is in transient state. This transient does not admit simple description, as it incorporates several mechanisms starting exactly when a key is pressed. Most such mechanisms produce audible onsets until the soundboard is loaded with sufficient energy and the steady state is reached. Their audibility before the soundboard takes the lead hence defines a transient soundfield, that can be characterized only if the instrument components responsible for these onsets are clearly localized.

Characterizing a transient soundfield is key for understanding whether a piano emits localized onsets, capable of activating the auditory precedence effect [5]. It is known that this effect locks the perceived localization of a sound source to the angular position from where sufficient energy has arrived to the ears during the first milliseconds of a sound event. If this were true for piano tones, then the subsequent steady-state transformation of the acoustic field in a set of frequency-dependent sound sources would have no chance to unlock the perceived angular position of a piano tone. The question, then, becomes to understand if the onsets forming the transient soundfield of a tone radiate approximately from the position of the key producing the corresponding note. To our knowledge, the most informative study on piano transients has been made by Askenfelt in 1993 [6]. In this study it was shown that the so-called touch precursor, which accounts for 20-25 ms of sound coming from the instrument *before* the hammer hits the string, contains in particular a distinct low-frequency “thump” radiating from the keybed. This noise originates when a key bottom bumps against the keybed wooden structure at the end of its fly. Such a bump can be roughly classified as a point-wise impact on a rigid part of the piano, hence potentially eligible for enabling the precedence effect. Yet, it is not safe to claim that a tone can be localized below the corresponding note key thanks to this process. In fact, the mechanical energy propagates much faster in rigid materials than on air; for this reason it must be reasonably assumed that pianists receive a relatively wide wavefront consequence of this bump instead of a well localized wave onset precurring the distributed emission of noise from the keybed.

2. MOTIVATIONS AND HYPOTHESIS

Fig. 1, previously published [7], seems to confirm the above assumption. It shows eight pressure signals measured in July 2012 inside a silent chamber at Viscount International SpA in Mondaino (RN), Italy, over the keyboard of a Seiler model 1849 playing a note C4, using a Bruel&Kjær model 4188 omnidirectional microphone array calibrated and made available by Angelo Farina’s research group. Considering that microphone channels 8 to 22 ranged from C2 to C6, hence covering four octaves, the plots in fact show too little relative delay among signals around microphone channel 14, which was closest to the key C4. Hence, this sound should not enable the precedence effect.

In the same publication it was shown that even a weaker hypothesis did not hold. In fact, pianists who were passively listening to a sound reproduction of the above eight channels scored scrambled versions of the same reproduction to be equivalently realistic in both sound and auditory scene quality [7]. Scrambling was obtained by re-wiring the input-output connections between channels into alternative configurations. Now, only the heaviest reconfigurations (e.g., random re-wiring or swapping of channel pairs between the left and right quartet) were scored significantly lower.

2.1 Active listening

So, why digital piano tonmeisters find consensus by customers (including skilled musicians) when designing piano tones which render definite localization cues? We approached the question by hypothesizing that — partially quoting [7] — independently of the existence of a precedence effect the localization of a note during playing could be locked to the corresponding key position by somatosensory cues of hand proprioception. Holding this “somatosensory precedence”, then the same position could be robustly recalled during listening: this previously learnt process may overwhelm or even suppress the auditory localization of the same note via lateralization cues, in particular resolving any potential incongruence between proprioceptive and auditory information.

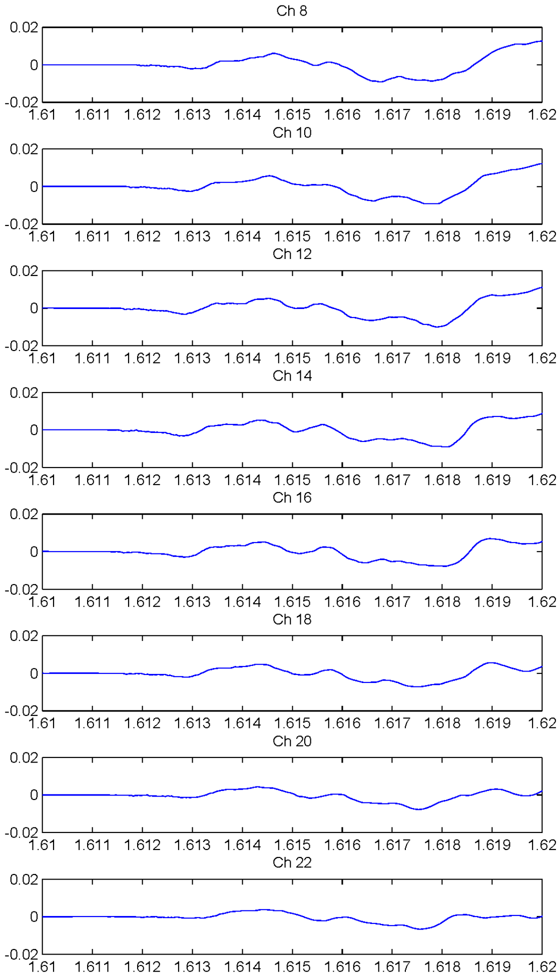


Figure 1. Temporal plots (10 ms) containing the attack of a C4 note on eight selected channels of a microphone array spanning four octaves (C2 to C6) of a Seiler 1849 piano keyboard. Microphone on channel 14 is closest to the key C4 [7].

While a role of the proprioceptive memory in support of the localization of passively listened piano tones is not hypothesized in the following, we experimented on the influence of active listening on the same process. In a previous experiment, whose outcome has been not yet systematically analyzed and for this reason it is still waiting to be published [8], pianists were asked to listen to the multichannel piano tones seen above. The only difference in the stimuli was that we used fourteen- instead of eight-channel reproduction of those tones, hence achieving a higher quality. Concerning the listening conditions, pianists attended also a session in which they actively played the notes instead of just passively listening to the corresponding tones. At each trial, every pianist rated the realism of the tone’s spatial presentation. The result, summarized by the histograms in Fig. 2 which are still waiting for an analysis of their exact significance, tells that pianists preferred presentations where the channels were not scrambled, in both

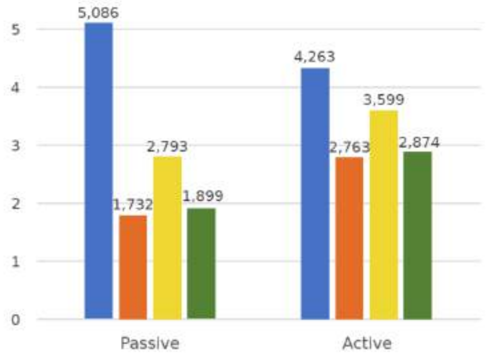


Figure 2. Histograms showing the perceived realism of tone presentations for passively (left) vs. actively (right) listening performers, respectively for unscrambled (blue), pan-reversed (red), randomly scrambled (yellow), and monophonic (green) array listening.

passive and active listening conditions. Furthermore, active listening supported the spatial realism of scrambled presentations.

In spite of its potential insignificance, this result is interesting as it suggests a logical cross-modal process at the base of spatial perception in pianists: if tones were perceived to sound realistic in space, then the auditory process was not supported by the kinesthetic action of playing the keys; if tones did *not* sound realistic in space, then the kinesthetic action conversely supported the same process. So, at least preliminarily, this experiment could be in favor of a key position-dependent auditory localization hypothesis, supported by somatosensory cues if the auditory process becomes confusing.

2.2 Current hypothesis

We investigated whether listeners localized tones on the corresponding keys. Binaural listening was enabled through headphones. Furthermore, on top of somatosensory feedback we tested also the role of visual feedback, by including in the experiment a condition in which participants while listening to a tone could see the corresponding piano key to move down. The test was targeted to individuals who were not pianists, to minimize the effects of repeated exposition to auditory feedback cues of piano tones following by long-practiced kinesthetic actions of key pression, and by the related somatosensory and visual perceptions.

3. SETUP

3.1 Recordings

As a preliminary step to the creation of suitable experimental stimuli, we made an extensive campaign of binaural recordings of piano notes. A grand piano Yamaha Disklavier (model DC3 M4) was used, equipped with sensors and actuators on keys and pedals which can be accessed via MIDI. The piano was placed in a laboratory space, therefore recordings contained also the room response: we do not consider this to be a limitation of the study, as we were interested in investigating ecological listening conditions. In order to avoid strong reflections from the walls, the piano was placed at the center of the room.

The recording setup is shown in Fig. 3 (left) and employed a KEMAR mannequin [9] with microphones placed at the entrance of the ear canal. As shown in Fig. 3 (right) the KEMAR mannequin was placed at the position of the pianist. The setup was moved to the center of the room. The height of the head was chosen to match that of one of the experimenters when sitting at the piano stool, resulting in stool height of about 45 cm from the ground. Additional binaural recordings were taken for different placements of the mannequin, although only frontal recordings taken at the pianist position are used in the remainder of this paper.

The two signals transduced by the mannequin were amplified first by the KEMAR 12AL preamps and then further amplified and sampled at 96 kHz and 24 bits through an RME Fireface 800 audio interface, which in turn was connected to a laptop. A SuperCollider patch[1] running on the laptop performed two tasks: (1) triggering the production of single tones on the Disklavier (through MIDI events), and (2) recording the binaural signals and storing them with appropriate names.

All the 88 keys were recorded. For each key, 10 key velocity values were recorded, from 12 to 111 with 11-step increments. These values were chosen for consistency with a previous study [10] where we collected vibration signals at all the piano keys for the same velocity values: the long-term goal is to develop a multimodal dataset of stimuli containing both binaural recordings and vibrotactile stimuli. While earlier recordings of vibration signals lasted 16 s [10], in this case it was found that this duration is not always sufficient, as the decays for the lowest keys at the highest velocities last much longer (up to 30 s). Therefore, variable recording times were chosen empirically depending of the key and the velocity value, with the goal of reducing the total time of a recording session (which lasted as long as six hours). Recording times for single keys range from 30 s for A0 at velocity 111, to 12 s for C8 at velocity 12.

Recording sessions were performed automatically at night time. This choice had a twofold motivation: first, it was a necessity in order to avoid clashes with other laboratory activities carried out during the day; second, it served the purpose of minimizing environmental noise coming from the exterior (the lab is in a pedestrian-only university area, which is noisy during the day and extremely silent during the night). To this end, we used the launchd service-management framework to temporize the execution of the SuperCollider patch.

3.2 Playback

The setup for the playback of recorded binaural samples employed a laptop, connected on one side to a RME Fireface 800 audio interface and to a pair of Sennheiser HD600 headphones, and on the other side to the Yamaha Disklavier. Fig. 4 depicts the scheme of a SuperCollider patch that

[1] http://supercollider.github.io/

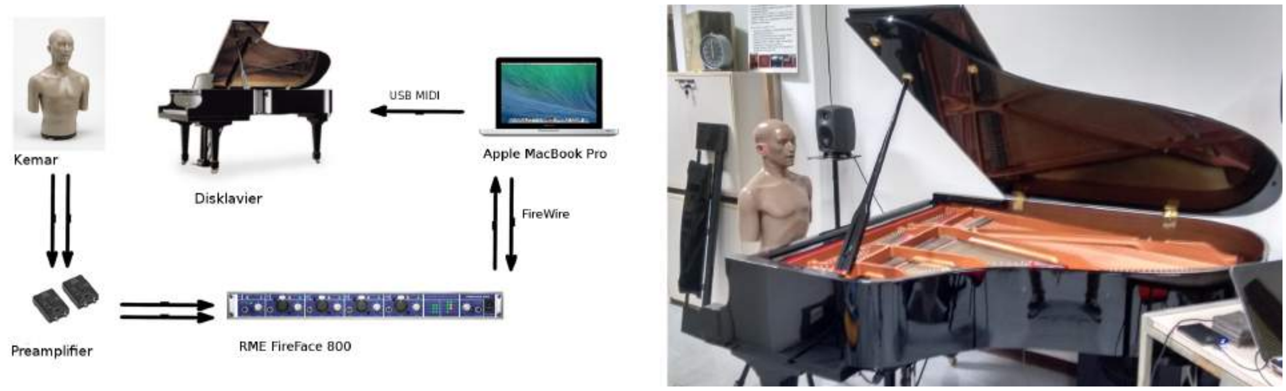


Figure 3. The recording setup. Left: block scheme of the acquisition chain. Right: a picture of the KEMAR mannequin positioned at the pianist location. The setup was moved to the center of the room for the recording session.

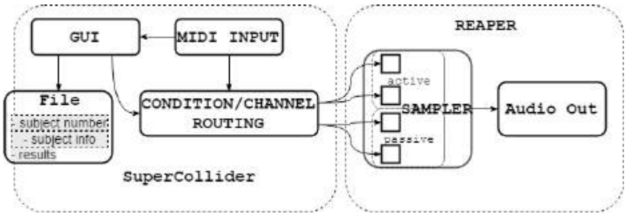


Figure 4. Scheme of the software employed for passive and active playback of binaural samples.

controls the playback conditions. Specifically, in a “passive” condition a binaural sample was played back through headphones, and the corresponding key on the Disklavier was simultaneously depressed by the actuator through a MIDI event. Conversely, in an “active” playback condition the patch received MIDI messages from piano keys depressed by the pianist, and played back the corresponding binaural sample through headphones. In all cases the playback of samples was delegated to a VST sampler hosted by Reaper[2] which in turn was controlled by SuperCollider.

Note that Disklavier pianos can be switched from normal operation to a “silent mode”, which disconnects the string board while leaving the MIDI features active. In this modality the hammers do not hit the strings, and therefore the instrument does not sound, while the remaining mechanical operations of the keyboard are left unaltered. Both active and passive playback conditions employed the Disklavier in silent mode, in such a way that no acoustic piano sound was ever superimposed to the recorded samples.

The Sennheiser HD600 are open headphones. We chose to use open headphones to provide subjects with ecological listening conditions where environmental sound was not masked. Moreover, these headphones have a very flat frequency response, which does not disrupt spatial information contained in the binaural signals. This was verified by comparing some of the recorded samples with the same samples played back through headphones and again recorded by the KEMAR wearing headphones. Pairwise comparisons showed very low spectral distortion. In light of this, it was decided that no equalization filter was needed to compensate for the headphone frequency response.

A calibration procedure was devised to match the loudness of the samples played back through headphones to that of the true Disklavier sound at the pianist position. Again, the samples were played back through headphones and recorded by the KEMAR wearing headphones. Comparing the loudness of these re-recordings to that of the original ones led to the insertion of a $+18$ dB gain. Such a high gain had the drawback that the background noise became perceptible in the late stage of the note decay. However this problem was mitigated by the fact that experimental stimuli lasted only 5 s (see Sec. 4 below), and by the use of open headphones which let environmental sound go through.

4. EXPERIMENT

In order to investigate the role of the auditory, visual and somatosensory feedback in piano tone localization, we devised an experiment in which these three types of information were combined in different conditions.

4.1 Stimuli

Three listening conditions were used. In all of them the subject was sitting on the piano stool at the pianist’s position in front of the Disklavier; the lid of the piano was fully raised as during the recordings. The first condition employed passive listening of binaural samples with the fallboard closed (“passive closed” condition hereafter). The second one again employed passive listening of samples, but this time with the fallboard open in such a way that the subject could see the movement of the corresponding key (“passive open” condition hereafter). Finally, the third condition employed active listening of samples, in which the subject actively pressed a key on the piano and heard as a consequence the corresponding sample (“active” condition hereafter).

A second factor was channel routing, for which two settings were used. In the first one (“standard” setting hereafter), the two channels of the frontal binaural recordings were routed to the correct headphone channels, while in

[2] http://www.reaper.fm

the second one ("reverse" setting hereafter), they were reversed so that the left channel was routed to the right headphone channel and viceversa.

Experimental stimuli were then constructed using only a subset of the A tones, namely the five tones A0, A1, A3, A6, A7. This choice was driven by the need of keeping the total duration of a single experimental session to a reasonable time.

The duration of the stimuli was limited to 5 s, for the same reason. Additionally, 5 s long stimuli minimized the perceptibility of background noise in the late decay stage of the samples, as discussed in Sec. 3 above. In the active condition, the 5 s stimulus duration was enforced by showing a timer to the subjects and instructing them to release the key after this duration; if the key was not released after this duration, a note-off event was automatically sent.

The total number of trials was 120 (i.e., 5 samples × 2 routing settings × 4 repetitions × 3 conditions), thus 40 for each of the three conditions. In the passive closed and passive open conditions the key velocity was set to 84 in all samples. In the active condition the key velocity was set by the playing subject, controlling the corresponding sample play back.

4.2 Procedure

Subjects with an even ID attended the active, passive closed and passive open condition in sequence; those with an odd ID attended the passive closed, passive open, and active condition instead. Stimuli were randomized within each condition.

After each trial, the subject had to rate the stimulus along two dimensions: the overall "sound quality", rated on a 7-points Likert scale (from 1 to 7), and the perceived "lateral direction" of the sound, again on a 7-points Likert scale (from -3 to $+3$). Subjective ratings were recorded through a graphical interface on a laptop placed at the piano music stand. The interface was composed of a vertical slider for the sound quality ratings and a horizontal slider for the lateral direction ratings. The same interface guided subjects through the experiment. For the active condition, the actual (MIDI) note values and velocity values of the keys played by subjects were recorded as well.

Experimental sessions started with a brief verbal introduction to the subjects, who were told that the goal of the experiment was to evaluate the "quality" of piano sound, with no mention of localization or spatial features of the sound. Then the various phases of the experiment were explained. In particular, subjects were instructed to keep their head still during each trial and to maintain the visual gaze on a single location in front of them. This ensured that, during the playback of the binaural stimuli, the reference frame of the subjects' heads remained coherent with that of the Kemar head (with which the same binaural stimuli were acquired).

Then, a pre-experimental interview was carried out to collect age, self-reporting of hearing problems, degree of familiarity with the acoustic piano as an instrument. Three alternative answers to the last question were given: performer, listener, no familiarity (meaning that the subject had never experienced the sound of an acoustic piano in live conditions). During the session, the experimenter annotated relevant reactions or behaviors by the subjects. Finally after the session an informal post-experimental interview was carried out with questions related to musical competencies of the subject and her/his impressions of the experiment.

Experimental sessions lasted about $35-40$ minutes. A total of 13 subjects took part in the experiment (age $17 - 26$ years, average 22.2). None of them was a pianist. Interviews revealed that 8 of them were not familiar with the acoustic piano, 8 were listeners, and 1 was a performer although not at a professional level. Moreover, 6 subjects had no music training, 2 had practiced a musical instrument as amateurs for a short amount of time, and 5 received music training and played a musical instrument although not at a professional level.

4.3 Results

We analyze and discuss results regarding the second dimension rated by subjects, i.e. perceived "lateral direction" of the sound. One-way ANOVAs with factor Condition were performed separately for each note, and for the two routing settings independently. Such a choice is preliminary at this stage of the research, since clearly it prevents from bringing to surface the interactions among factors. Even without analyzing such interactions, the results we found using individual ANOVAs are sufficiently rich to lead to an articulate discussion.

Fig. 5 shows boxplots for the standard routing setting. Physical positions of the five A keys have been normalized in the range [-3,3], see Sec. 4.2. Furthermore, perceived note position means are reported for each of the three listening conditions (passive closed, passive open, active). The result is mixed: lower tones were localized on positions having clear relationship with the keys; localization, though, ceased to depend on key positions for the higher notes, which in fact were localized slightly leftward from the center.

In the reverse routing setting, similar boxplots were found. Fig. 6 in fact shows that participants located the tones assimilably on reversed positions, again displacing the higher notes to the wrong half of the keyboard.

For each note a one-way ANOVA was conducted among the three listening conditions, separately for the standard and reverse settings. The ANOVA showed significant differences between conditions for note A1 with the standard setting ($F(2, 140) = 6.15$, $p < 0.01$), and for note A7 with the reverse setting ($F(2, 125) = 5.25$, $p < 0.01$). Pairwise post-hoc t-tests for paired samples were ran, using Holm-Bonferroni correction on p-values. The post-hoc analysis revealed that note A1 was localized in significantly different positions in active vs. passive open conditions ($t(89) = 3.52$, $p < 0.01$), and in active vs. passive closed conditions ($t(89) = 3.17$, $p < 0.01$). The same analysis showed that note A7 was localized in significantly different positions in active vs. passive closed conditions ($t(74) = 3.09, p < 0.01$), and in passive open vs. passive closed conditions ($t(102) = 2.34, p < 0.05$).

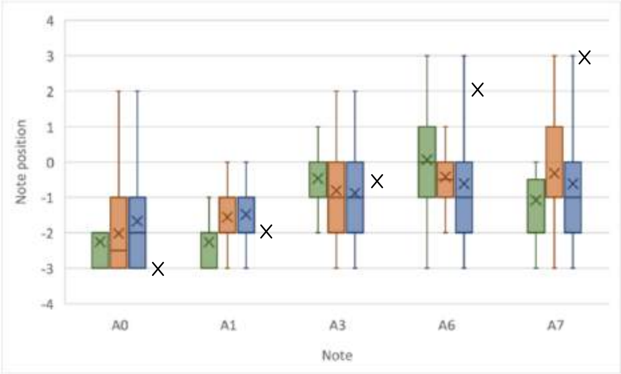


Figure 5. Key vs. perceived note position for the standard routing setting. Right to left boxplots for each note: perceived position in the passive closed condition; perceived position in the passive open condition; perceived position in the active condition. Note positions are normalized in the range [-3,3] and marked with 'X': A0=-3; A1=-2; A3=-0.5; A6=2; A7=3.

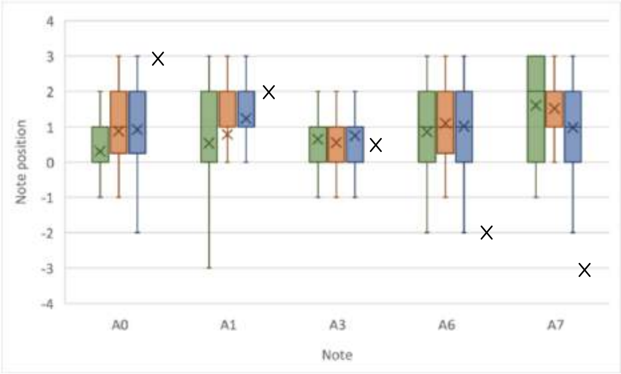


Figure 6. Key vs. perceived note position for the reverse routing setting. Right to left boxplots for each note: perceived position in the passive closed condition; perceived position in the passive open condition; perceived position in the active condition. Note positions are normalized in the range [-3,3] and marked with 'X': A0=3; A1=2; A3=0.5; A6=-2; A7=-3.

A further series of one-way ANOVA was conducted for each condition and routing setting, this time to uncover significant differences among notes. These ANOVA showed that all notes were localized on significantly different positions and routing setting, except in the case of passive closed condition with reverse setting. Pairwise post-hoc t-tests for paired samples were ran, using Holm-Bonferroni correction on p-values. The post-hoc analysis revealed a number of significantly different pairs. For brevity, individual t-statistics are omitted and only the significant p-values are listed in Table 1.

5. DISCUSSION

First of all, with standard routing setting listeners did localize all tones without significant support (except for note A0) from the visual or somatosensory feedback. Notes A0, A1 and A3 were localized around the corresponding key positions: in this case the visual and somatosensory feedback in general helped refine the localization, meanwhile with apparently stronger role of the somatosensory than visual feedback in locking the sound source in correspondence of the respective key. The data are also less dispersed when somatosensory perception was enabled. On the contrary, notes A6 and A7 were localized off key position. In this case the contribution of the visual and somatosensory feedback to the auditory localization is not clear: for note A6, the two sensory modalities seemed to shift the sound source position toward the key; for note A7, they probably wandered around the sound source position where it was localized through listening.

	standard			reverse		
	A	C	O	A	C	O
A0-A1						
A0-A3	●	○	●			
A0-A6	●	●	●			
A0-A7		●	●			
A1-A3	●		●			
A1-A6	●	○	●			
A1-A7	●	○	●			
A3-A6						○
A3-A7				●		●
A6-A7						

Table 1. Significantly different note pairs under different listening conditions and routing settings. A: active; C: passive closed; O: passive open. '○': $p < 0.05$; '•: $p < 0.01$'.

In the reverse routing setting, listeners appear more confused. Notes are largely localized in the normalized portion [0,2] of the keyboard range. Once again there was no significant support from the visual or somatosensory feedback, except for note A7 which, however, suffers from the most inaccurate localization. In general the somatosensory data are more dispersed, suggesting that active playing did not help reduce confusion in subjects when listening to reverse routing.

In the limit of its significance, the ANOVA among conditions suggests the existence of a progressively supportive role of the visual and then somatosensory feedback in localizing a tone over the corresponding key when the auditory system did not contradict this localization process. Conversely, such two sensory modalities would have no decisive role when the auditory localization failed to match the sound source with the corresponding key position.

The same analysis was repeated for the amateur musicians subgroup (5 subjects), and for subjects with an even and odd ID who were respectively exposed to a different order of the experimental conditions—refer to the beginning of Sec. 4.2. Results were analyzed informally, due to insufficient population in those subgroups. Amateur musicians provided answers that are less dispersed around the note position, whereas they did not perform differently from non-musicians under reverse routing conditions. The same effect seems to affect subjects with an even ID, i.e., those who attended the active condition first. A previous knowledge of music playing, then, could have supported subjects in making more correct decisions in normal routing conditions. Similar support could have helped those subjects who first played the notes during the test.

The ANOVA among note positions did not contradict the previous analysis among conditions, and suggests further observations. First, in the reverse routing setting there was certainly more confusion in localizing the tones on significantly different positions of the keyboard. Visual feedback was relatively more decisive in this sense, not only with the reverse but also with the standard routing setting. In this case tone localization was more reliable: Table 1 shows a generally increasing role of the somatosensory (A), auditory (C) and visual (O) modality in supporting the localization of the notes on different keyboard positions. One would be tempted to sort the rows in that table in key distance order, and expect to find an increasingly different perceived position of the tones based on this order. Unfortunately this is true only to a limited extent: pairs A3-A6 and A3-A7, for instance, which fall about on the middle rows if the table is reorganized in key distance order, did not give rise to distinct perceived positions in the keyboard under any type of feedback.

A comparison between our analysis and the data in Fig. 2 does not permit to align this discussion to one possible result suggested by that figure, i.e., that somatosensory feedback may have supported tone localization on the corresponding keys when the auditory feedback was unreliable; in fact, our experiment mainly suggests that visual and somatosensory localization worked better if not contradicting the auditory localization. This misalignment indeed could be expected, since we switched from loudspeaker to headphone listening and furthermore we experimented with participants who were not pianists.

6. CONCLUSIONS

An experiment has been made to investigate the role of the auditory, visual and somatosensory feedback in piano tone localization. We inherited previous literature about the characteristic transients precurring a steady tone in the piano, along with yet partially unstable findings about spatial realism of piano tones using auditory and somatosensory cues. Moving from this literature, we hypothesized that somatosensory and visual feedback supported the auditory localization of piano tones on the corresponding keys. Overall the results suggest that this happens in the cases when the auditory feedback is coherent with the somatosensory and visual feedback, in the sense of the hypothesized specific localization process. However, these results are difficult to be compared with the previous findings.

Given also the different hypothesis as well as conditions existing between the previous tests and the experiment proposed here, we remain noncommittal about the existence of an auditory precedence effect leading the localization of piano notes, whose key positions are well-known by pianists in any case through their visual and somatosensory experience. Rather, we have collected results under a broad set of conditions which may prove precious to build future experiments. Such experiments, first of all, should include a systematic search for robust acoustic precursors that may be responsible for the activation of the precedence effect.

7. REFERENCES

[1] R. D. Lawson, “Loudspeaker system for electronic piano,” US Patent 5 789 693, Aug., 1998.

[2] K. Yamana, “Electronic musical instrument having a pan-pot function,” US Patent 4 577 540 A, Aug., 1986.

[3] N. H. Fletcher and T. D. Rossing, *The Physics of Musical Instruments*. New York: Springer-Verlag, 1991.

[4] S. Koseki, R. Mantani, N. Sugiyama, and T. Tamaki, “Method for making electronic tones close to acoustic tones, recording system and tone generating system,” EU Patent EP1 357 538A3, Apr., 2003.

[5] C. Tsakostas and J. Blauert, “Some new experiments on the precedence effect,” *FORTSCHRITTE DER AKUSTIK*, vol. 27, pp. 486–487, 2001.

[6] A. Askenfelt, “Observations on the transient components of the piano tone,” *STL-QPSR*, vol. 34, no. 4, pp. 15–22, 1993, web available at http://www.speech.kth.se/qpsr.

[7] F. Fontana, Y. D. Pra, and A. Amendola, “Sensitivity to loudspeaker permutations during an eight-channel array reproduction of piano notes,” in *Proc. SMAC/SMC 2013*, Stockholm, Sweden, Jul. 30 - Aug. 3 2013.

[8] F. Fontana, S. Zambon, and Y. D. Pra, “Designing on subjective tolerance to approximated piano reproductions,” in *Proc. of the Third Vienna Talk on Music Acoustics*, Vienna, Austria, Sep. 16-19 2015, pp. 197–204, invited paper.

[9] W. G. Gardner and K. D. Martin, “HRTF measurements of a KEMAR,” *J. Acoust. Soc. Am.*, vol. 97, no. 6, p. 3907–3908, Jun. 1995.

[10] F. Fontana, F. Avanzini, H. Järveläinen, S. Papetti, F. Zanini, and V. Zanini, “Perception of interactive vibrotactile cues on the acoustic grand and upright piano,” in *Proc. Joint ICMC/SMC Conf.*, Athens, Greece, 2014, pp. 948–953.

SONIC GREENHOUSE - Considerations on a Large-Scale Audio-Architectural Installation

Otso Lähdeoja
University of the Arts Helsinki Finland,
MuTri / Music Technology Department
otso.lahdeoja@uniarts.fi

Josué Moreno
University of the Arts Helsinki Finland,
MuTri / Music Technology Department
josue.moreno.prieto@uniarts.fi

ABSTRACT

This article discusses a recent large-scale audio-architectural installation which uses the glass structures of a greenhouse to create a multichannel sound system driven by structure-borne audio transducers. The sound system is presented and its implementation is discussed in reference to the constraints of a site-specific installation. A set of sound spatialisation strategies are proposed and their effectiveness weighted in the specific context of a large-scale work where the audience acquires an active role in moving within the piece, countering traditional centered spatialisation methods. Compositional strategies are developed in response to the context, emphasizing the spatial dimension of composition over the temporal and narrative ones and pointing towards the concepts of "Sonic Weather" as well as "Sonic Acupuncture".

1. INTRODUCTION

This article constitutes a report and an analysis of a large-scale audio-architectural installation "IN SITU: Sonic Greenhouse", which took place in September-October 2016 at the historical Winter Garden Greenhouse in Helsinki[1], Finland, and was experienced by approximately six thousand visitors. The piece's rationale is to transform the glass structure of the greenhouse into a multichannel sonic architectural object - or a macroscale musical instrument - by activating the glass panels with an array of sixty structure-borne sound drivers, transforming them into visually transparent loudspeakers. The idea is to drive sound all over the greenhouse's structure, creating an immersive aural architecture at the crossroads of software-controlled multichannel sound spatialisation and a soundscape inviting for a first-person discovery.

The work brings together a combination of sound and architecture with the larger issues related to the "greenhouse"-concept, such as environmental issues and the poetics of connection/separation inherent to a glass structure. A greenhouse is a man-made biotope where favorable conditions are created in a delimited space for plants which would not survive in the local natural conditions. The greenhouse metaphor was developed in the compositional and spatialisation strategies involved in the piece, as well as in the establishment of a weather data stream into the installation's generative sound engine. Sonic Greenhouse was a collaborative work by two composers of electronic music. The process was completed in parallel and in dialogue, with no specific personal tasks assigned. The resulting piece can be considered to be a duo. The piece's website can be accessed at: http://sonicgreenhouse.eu.

In this article we discuss examine the piece from three different interrelated levels of analysis: 1) sound diffusion system design and implementation, 2) spatialisation and data input, and 3) compositional strategies.

"Sonic Greenhouse" is a work stemming from both artistic and technological research, aiming for the fertile dialogue between the two. The technical research motivations involved testing a large-scale structure-borne sound diffusion system directly on the structure of a building. We had no precedent knowledge of how such a system would sound and how it could be used to produce effective spatial aural impressions. On the artistic side, our research questions related to finding the optimal compositional strategies for a site-specific work in a large space with defined architecture, luxuriant plant life, social behaviour and interaction, abounding with multisensorial stimuli.

2. RELATED WORK

Creative combinations of audio and architecture has been explored in a number of works, connecting "Sonic Greenhouse" to a continuing line of aural installation art. Building structures have been used as sound sources, as in David Byrne's piece "Playing the building": "Playing the Building is a sound installation in which the infrastructure, the physical plant of the building, is converted into a giant musical instrument. Devices are attached to the building structure — to the metal beams and pillars, the heating pipes, the water pipes — and are used to make these things produce sound. The activations are of three types: wind, vibration, striking. The devices do not produce sound themselves, but they cause the building

[1] Talvipuutarha is the Helsinki City public greenhouse, open for visits since 1907. http://www.hel.fi/hki/hkr/fi/Viheralueet/Talvipuutarha

elements to vibrate, resonate and oscillate so that the building itself becomes a very large musical instrument." [1]. "Sonic Greenhouse" presents a parallel with Byrne's piece in its aim to induce vibration into the building's physical structure, with the notable difference of the present project using audio-rate actuators rather than sub-audio frequency mechanical devices such as motors and solenoids. Audio-rate excitation at bass frequencies has been investigated within the "Resonant Architecture" project's different instantiations, exciting the lower spectrum resonant modes of large-scale architectural and sculptural entities [2]. Structural vibration has been used as a compositional element in numerous works of Bill Fontana, for example in the "Harmonic Bridge" piece at Tate Modern, London [3]. More recently, Rodolphe Alexis has used the Greenhouse context for sound installation, using wavefield synthesis to create a soundscape for a flower exhibition at Natural History Museum in Paris [4]. The present work builds on prior research on structure-borne sound in intermedia context by Otso Lähdeoja [5].

The compositional considerations of "Sonic Greenhouse" have echoes in Janet Cardiff's "Forty Part Motet", which sets the listener as the active agent in the unraveling of a spatially distributed audio work, leading into a first-person spatialisation [6]. A similar idea has been explored in "Parabolic Dishes" by Bernhard Leitner creating spatial regions sonically, in which the visitors are required to explore the acoustically structured space [7]. In a situation where the visitor is an explorative agent, questions of form and sound spatialisation strategies arise. In our work, this was reflected in a dialogue considering placing sounds in space rather than in time, and creating a generatively changing sonic situation rather than a narrative, echoing Max Neuhaus: "Traditionally composers have located the elements of a composition in time. One idea which I am interested in is locating them, instead, in space, and letting the listener place them in his own time." [8]

3. DESIGN AND IMPLEMENTATION

The Helsinki Winter Garden comprises three different spaces, as shown in Fig. 1: a) the central Palm Room (surface 280m^2), b) the eastern wing's Cactus Room (200m^2) and c) the western wing's social space (170m^2). "Sonic Greenhouse" was designed in intimate connection to the given space in order to foster integration into the physical structures, plan and human life, acoustics and poetics of the space. A prior period of sensorial research was completed by multiple visits to the site spanning over a year, observing how the staff and visitors behave in the space and how the building reacts to different weather conditions. These observations had a decisive effect on the design of the piece, which had to take into account high humidity levels, leakages and the regular activities such as watering the plants. The visits also served to familiarise ourselves with the acoustic specificities of the site.

The Palm Room (a) acoustics are very reverberant, cathedral-like and the space is filled with luxuriant vegetation. The sound work was designed to respond to this grandeur with full-spectrum and dynamic sonic constructions involving precisely rendered movements of sound within the glass structure. The Cactus Room (b) is very silent and intimate. In accordance, the sound was designed to occupy the space with precise, intricate sounds which draw the visitor's attention to details and silent contemplation. The social space (c) was left as a relaxation space, convenient for discussion and studying documentation about the work presented in printed format.

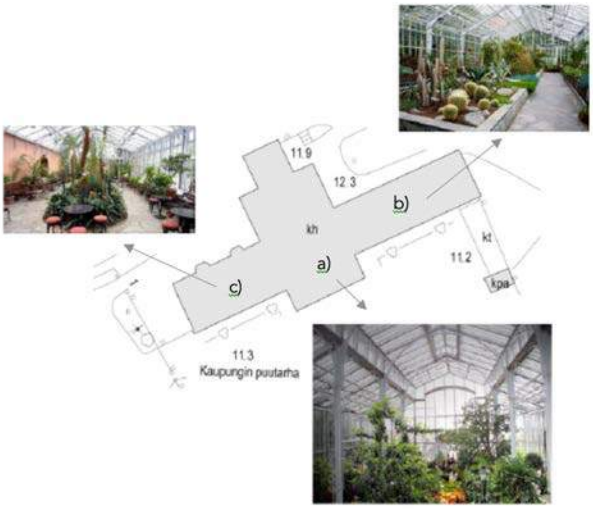


Figure 1: Helsinki Winter Garden floor plan and photos of its three rooms: a) The Palm Room, b) the Cactus Room and c) the Chatting Room

3.1 System Description

The Sonic Greenhouse implemented a sound diffusion system in two of the Winter Garden's three rooms. A 40-channel system was built in the Palm Room and another, independent system with 20 channels was built in the Cactus Room.

Two different types of audio transducers were used: 1) TEAX32C30-4/B structure-borne sound drivers attached on the building's glass walls and on custom-made suspended plexiglass panels, and 2) small cone loudspeakers enclosed in glass jars placed on the ground amidst the vegetation. Sound diffusion via structure-borne sound drivers was an aesthetic starting point for the entire piece. These drivers - or vibration speakers - enable to transform a rigid surface into a full-range speaker[2] with minimum visual impact. The driver and glass panel combination is acoustically ideal for creating a large-scale soundscape - oriented aural impressions. The panels act as flat panel dipole speakers, providing a larger and more dif-

[2] The measured frequency range of the TEAX32C30-4/B transducer is 70Hz - 14kHz.

fuse acoustic image than a more directive cone speaker. In this sense, diffuse sound emanating from glass panels creates an aesthetic unity: the sound source is not easily located, and perceptually it blends into the glass, its transparency and the surrounding landscape. The cone loudspeakers were designed to provide a spatial counterpoint to the wall and ceiling transducers, as well as to create a symbolic visual element - a speaker enclosed in glass replicates the greenhouse idea in miniature scale. In a sense the speaker jars work as a metaphor of the whole installation. The building is sounding through its glass and the speaker cones sounds are contained by the glass making the setting for an inner-outer sonic and visual interplay.

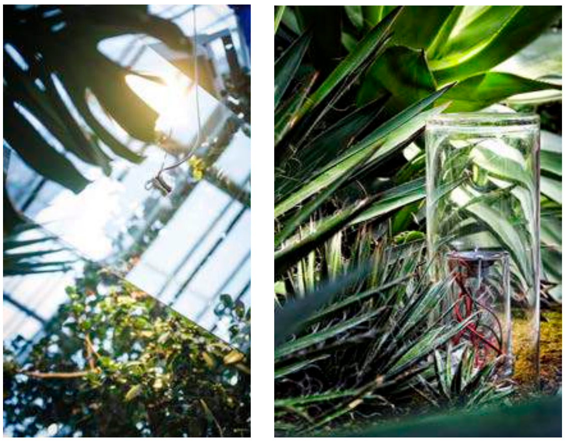


Figure 2: photos of transducer on glass panel and miniature speaker placed in a glass jar.

The audio transducer distribution in the space resulted from a compromise between the aim to produce a highly immersive sound field and the technical constraints of the building. Immersive diffusion of sounds from the ground, walls and ceiling was a priority in our design. However, the building's top layers of glass were not accessible from the inside with a scissor lift. Also, attaching panels amidst the vegetation was not feasible due to plant protection regulations. With these constraints, we designed a hybrid system of glass-panel mounted transducers, plexiglass panel speakers and loudspeaker jars, all characterised by visual transparency and a unified aesthetic design criteria.

3.2 Audience Feedback

The installation was open to the public for two weeks in September-October 2016 and it was well visited - the estimation of reached audience is 6000 persons. The Helsinki Winter Garden is a popular recreation place for locals as well as tourists, and many visitors did not come specifically for the installation, rather for the site itself. The site also has a strong clientele of people who visit regularly, even almost every day (such as elderly people living nearby). Audience reactions were overall positive, but varied strongly between the motives of the visitors. Some Winter Garden regulars felt that their beloved garden should not be altered in any way, some others were enthusiastic about the novelty and the added auditive dimension. The authors did not collect feedback in a systematic manner, but a guest book was available at the installation. A theme that arose from the guest book comments was the felt need for more explanatory material concerning the audio techniques involved which seemed too abstract for some visitors to appreciate. An intellectual counterpart to the piece's sensorial character could have been beneficial, such as written documentation, schemas and links to web media. We also conducted a final interview with the Winter Garden staff in order to have their impressions recorded. The staff's reactions were of particular importance for us, as they acted as project collaborators and they harbor a very intimate relationship both with the building and the plants that they take care of every day. The staff members were very enthusiastic about having a sound art premiere in the greenhouse, and they noted a revived interest in the site in some specific categories of audience such as elderly people, impaired persons and children. For the staff members, the sonic dimension created a novel dimension in a space they know very well, and they appreciated spending two full weeks discovering relationships between architecture, plants and sounds. By the end of the installation, it was agreed that two weeks was a correct duration – for a longer installation the sonic matter should be changed periodically in order not to become too redundant.

4. SOUND SPATIALISATION AND WEATHER DATA FEED

Two separate sound diffusion systems were implemented, one in the larger Palm Room and a second in the Cactus Room, due to the will to design two very different sonic atmospheres and spatialisation control systems. A common starting point for both systems was the determination to avoid any concept or technique of a sound field centered around a sweet spot. The Winter Garden's central areas are occupied by plants and there is no single central space available for audience. In addition, a typical visit to the Winter Garden involves walking from one room to the other, between plants, and a perceptual movement between large ensembles and tiny details. The sound spatialisation was designed from the following premises: moving audience, immersive soundscape allowing for perceptual zooming into details and out towards general impression, as well as a detailed, moving and lively general character.

Sonic Greenhouse stems from a collaboration between two composers of electronic music, each following a specific aesthetic agenda. In this setting, we chose to experiment with a set different and complementary approaches to sound spatialisation, ranging from the distribution of numerous multichannel sources up to the point of multichannel redistribution of a stereo source creating the multichannel perceptual sensation due to the extended size of the space. Working in collaboration created a dialogue for testing a range of ideas on sound diffusion. While Moreno had a very strong preference for avoiding panning trajectories, Lähdeoja was intrigued by the perceptual effects of large-scale sonic trajectories in such complex multi-channel acoustic space. The following sections provides a detailed account of the different spatialisation techniques involved in the piece.

4.1 The Palm Room (Main Hall)

For the Winter Garden's main hall - or Palm Room, each composer created three compositional modules, combined within a generative Max MSP mixer diffusing the modules in random order.

The spatialisation strategies were:

- Single sound source trajectories generated with distance-based amplitude panning (DBAP)[9], combined with larger masses of more stationary sonic material as spatial counterpoint.

- Polyphonic envelope-based texture in which every sound comes from only one channel. The panning sensation is produced by having similar sonic content on different channels.

- Sonic "breathing" in and out of the space: a sonic gesture would be initiated from a single diffusion point and then gradually spread to all channels, or the inverse gradual "retreat" process.

- Random "hard-panning" distribution: a sound is randomly switched between all channels in a temporal sequence.

- Stereo source spread across multiple channels: Channel location and surfaces alter the perception of pitch and timbre.

- One sonic element in the center while the other element travels from channel to channel exciting different areas of the site one at a time.

4.2 The Cactus Room: Weather Data-Driven Granular Engine

For the Cactus Room, our aesthetical decision was to associate sound particle clouds diffused at hearing threshold levels with the dry acoustics and desert-like atmosphere of the space. A 19-channel instance of the [munger1~] MSP external [10] was used as the basic grain engine, completed with Moreno's design of a generative wavy process based on pre-synthesized grain streams that add a complimentary mid-range frequencies to the overall spectrum.

Both granular engines and their source audio material were controlled by real-time weather data retrieved from the openweather internet service[3]. Wind velocity, temperature, pressure, and humidity data were mapped both to macro-scale events such as sound selection and mixing and to microsound-level operations like grain density and spectrum.

The motivation for connecting the Cactus Room's internal "sound life" to the external weather conditions stemmed from the poetics of transparency and frontier of a greenhouse. Within the glass frame one can visually observe outside events, but has a very limited perceptual access to the elements. By connecting the outside conditions to the sound engine, we envisioned to find a way around the greenhouse's "boundary condition". As a result, we obtained a perpetually variating generative sound structure tightly related to the atmosphere inside as well as outside the room.

4.3 Audience Agency: First-Person Spatialisation

While a considerable attention was invested into the prior design of sound spatialisation, once the installation was open to the public it became clear that the foremost spatialisation strategy was not computational, but rather the individual human trajectories within the piece. Writing on his piece "We Have Never Been Disembodied" at the Mirrored Gardens, Guangzhou, China, Olafur Eliasson echoes precisely our findings on how Sonic Greenhouse came alive by the individual agencies of the visitors: "For 'We have never been disembodied', I was inspired by the idea of using this humble context, intended for plants and agriculture, as a platform where full responsibility is handed over to the visitor, enabling him or her to become an agent in the space that is Mirrored Gardens. It is a platform of potentials, taking the intimacy of the village to its extreme, allowing for micro-sequences when visitors move through the building, and making explicit the temporal dimension of life." [11]

Eliasson's work has been analysed by Adam Basanta and extended to the conceptualisation of form in audiovisual installations, where "the three Euclidian dimensions (length, width and height) are not only modulated by the fourth, topological dimension (time), but also by the fifth 'dimension' of perceiver subjectivity. [...][12] Following Eliasson, we can contemplate form as the particular temporal experience of the first-person subject as they navigate in, through and out of the work's frame. That is, form as the particular first-person narrativisation of experience in a given installation." [13]

Basanta's analysis corresponds to the experiential outcome of Sonic Greenhouse. In the design phase, we did not predict the importance that the audience agency would take in the final work. It turned out that not only spatialisation, but also the form and narrative of the piece were largely defined by the first-person trajectory within the space and between the rooms. In large spaces with numerous non-symmetrically placed sound sources, the audience becomes the primary spatialisation agent. This finding has important consequences for our future designs of large-scale audio works, as it challenges estab-

[3] https://openweathermap.org/api

lished ways of designing sonic spatialisation as well as form in audio art.

5. COMPOSITIONAL STRATEGIES

5.1 Considerations on Time and Spatiality - Composing Sonic Weather

"I was to move from structure to process, from music as an object having parts, to music without beginning, middle, or end, music as weather" - John Cage [14]

Music is often considered as a time-based art. Max Neuhaus proposed that musical creation is not about placing sounds in time but rather in space, the visitor being the agent who would enact the time dimension. [8]. Audio artists who are notorious for creating an alternative use of time in their sound works (La Monte Young, Phil Niblock, a. o.), usually rely heavily in using either long sustained sounds or loops. In this work we propose the creation of a more complex sonic environment that surpases the perception of sonic discourse as a narrative in time, getting closer to a "sonic weather" echoing the words of John Cage. When listening to this sonic weather time becomes a secondary dimension.

When articulating sonically a complex public space, one of the possible strategies to follow is to create different areas of sonic colour in which the sonic material is grouped and spread creating a unique experience for every visitor, seeking to foster a bodily dialogue between the visitor and the space. This approach requires a non-linear narrative. in "Sonic Greenhouse", an algorithm controlled the continuous crossfading between predefined sonic situations achieving greater level of sonic complexity while keeping a natural blend of elements. This continuously evolving sonic complexity was not overwhelming to the visitors perception but rather contributing to the feeling of being immersed in a sonic weather, an intense experience of the space surpassing the perception of temporal discourse.

For "Sonic Greenhouse" we built an array of active and passive sonic objects to use in order to work as aural architects [15] creating a sonic weather focusing more in placing sounds in space rather than in time. In that sense, our aim as composers was to approach aural architectural composition in an atmospheric manner, favoring peripheral perception and light gestalt [16]. The piece illuminated sonically the architecture with a combination of active and passive sonic objects. The active sonic objects were the sound emitting glasses of the windows, the plexiglass panels, and the speaker cones inside the glass pots; while the passive objects were the glass pots which filtered the sounds coming from the speaker cones, the plexiglass panels that created a more complex network of acoustic reflections and spatio-morphologies [17], and the reverberation of the room.

In order to blend our sonic layer with the preexisting sonic structures and regular daily activities of the space we took a combined approach using a combination of recorded instruments —metallophones, organ, piano, and banjo, field recordings, and synthetic sounds, carefully tuned for the space and tested through regular visits. By tuning the sounds and imitating behaviour of preexisting sonic elements —like the wind in the surroundings or the water inside which were used as ghost electronics [18]— and by echoing the local weather conditions we created an organic soundscape that would create a continuity sensation avoiding to create any kind a sonic shock when entering or leaving the space.

5.2 A Note on Sonic Acupuncture

A possible definition of acupuncture can be: a local action by means of a pressure point on a key spot that has the power to change the situation globally, beyond the local area in which the pressure point is applied. Therefore, sonic acupuncture relies in applying sonic pressure points on key spots affecting the global sonic situation. These sonic pressure points can consist either on active acoustic objects, passive acoustic objects, or a combination of both. Urban Sonic Acupuncture parallels the practice of Urban and Public Space Acupuncture [19] in the Aural architecture field. Aural architecture deals with spatial and cultural acoustics, it also assigns four basic functions of sound in space: social, navigational, aesthetic and musical spatiality [15]. Artistic sonic interventions are placed along this axis by starting a negotiation between artistic intentions and the local knowledge and practices. "Sonic Greenhouse" proposed an early attempt of sonic acupuncture by means of glass speaker-pots placed in the central area of the Palm room. In some sections, these devices were emitting slowly sweeping frequencies sine waves[4] generated algorithmically altering the perception of the sonic weather at that given moment, also driving attention to the lower layers of the space and to specific plants in the area.

6. CONCLUSIONS

In this article we have discussed a recent large-scale audio-architectural installation from three points of view, namely system design, sound spatialisation and external data feed, as well as compositional strategies. The work was thought as a research process for a multichannel structure-borne sound system mounted directly on a building's structure, combined with specific aesthetic aims related to the Helsinki Winter Garden greenhouse. The results confirm a proof of concept: structure-borne sound drivers are an effective a meaningful way to sonify a large architectural space, presenting perceptual advantages in comparison to traditional cone speakers, such as radiation patterns and non-invasive appearance. Moreover, the results point towards a reformulation of concepts in sound spatialisation and form composition for

[4] Alvin Lucier - *"music for piano with slow sweep pure wave oscillators"* (1992). *"PIANOSCULPTURE for Mario Prisuelos"* (2016) is a piano and tape piece by Moreno based on the same principle.

large-scale works where the audience becomes an active agent in the space. Temporal and spatial elements unfold according to an individual exploration, which counter narrative designs thought for stationary audiences. Alternative compositional strategies were implemented, such as composing a Sonic Weather evolving in space rather than in narrative time, as well as Sonic Acupuncture meaning small-scale punctual sonic interventions in precise locations.

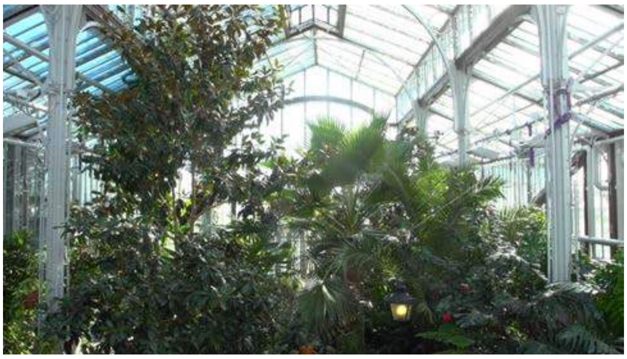


Figure 3: General view of the Palm Room, where 40 channels of sound diffusion were implemented.

Acknowledgments

Work on the Sonic Greenhouse project was supported by University of the Arts Helsinki Sibelius Academy Music Technology Department, Kone Foundation, Arts Promotion Centre Finland and the City of Helsinki. The prior research work in structure-borne sound and in composition was supported by Academy of Finland and Niilo Helander Foundation. The authors would also like to thank the Helsinki Winter Garden staff for their support.

7. REFERENCES

[1] D. Byrne, Playing the Building, http://journal.davidbyrne.com/explore/playing-the-building, 2005

[2] N. Maigret and N. Montgermont, The Resonant architectureProject: http://resonantarchitecture.com, 2009-2013

[3] Bill Fontana, Harmonic Bridge, http://resoundings.org/Pages/Architectural_Sound_Sculptures.html

[4] R. Alexis, Mille et une Orchidées, http://www.mnhn.fr/fr/visitez/agenda/exposition/mille-orchidees-2017

[5] O. Lähdeoja, “Structure-Borne Sound and Aurally Active Spaces” in Proceedings of New Interfaces for Electronic Music, NIME 2014, p. 319-322.

[6] J. Cardiff, 2001, The Forty Part Motet, http://www.cardiffmiller.com/artworks/inst/motet.html

[7] B. Leitner, P.U.L.S.E., Ostfildern: Hatje Cantz Verlag, 2008.

[8] M. Neuhaus, Max Neuhaus: Inscription, sound works Vol. 1 (p. 34). Cantz Verlag, 1994.

[9] T. Lossius, P. Baltazar and T. de la Hogue, DBAP–distance-based amplitude panning. Proceedings of the International Computer Music Conference, ICMC 2009.

[10] I. I. Bukvic et al., Munger1~: towards a Cross-Platform Swissarmy Knife of Real-Time Granular synthesis. Proceedings of the International Computer Music Conference, ICMC. 2007.

[11] O. Eliasson, We Have Never Been Disembodied, http://www.vitamincreativespace.com/en/?work=5607

[12] O. Eliasson, Vibrations. In O. Eliasson, I. Calvino, I. Blom, D. Birnabaum and M. Wigley (authors) Your Engagement has Consequences on the Relativity of Your Reality. Baden, Switzerland: Lars Müller Publishers, 2006.

[13] A. Basanta, “Extending Musical Form Outwards in Space and Time: Compositional strategies in sound art and audiovisual installations.” in Organised Sound, 2015, pp 171-181

[14] J. Cage, “An Autobiographical Statement”, in *Southwest Review, 1991*

[15] B. Blesser and L. Salter, Spaces speak, are you listening?: experiencing aural architecture. The MIT Press, 2007.

[16] J. Pallasmaa, Atmosphere as place. The Intelligence of Place - Topographies and Poetics. Edited by Jeff Malpas. Bloomsbury Publishing, 2014.

[17] D. Smalley, “Spectro-morphology and structuring processes. The Language of Electroacoustic Music.” in Organised Sound, 1986.

[18] H. Whipple, “Beasts and Butterflies: Morton Subotnick's Ghost Scores.” in The Musical Quarterly, 69(3), 1983. pp 425-441

[19] H. Casanova and J. Hernandez, Public Space Acupuncture. Strategies and Interventions for Activating City Life, Actar, 2015.

THE CSOUND PLUGIN OPCODE FRAMEWORK

Victor Lazzarini
Maynooth University,
Ireland
victor.lazzarini@nuim.ie

ABSTRACT

This article introduces the Csound Plugin Opcode Framework (CPOF), which aims to provide a simple lightweight C++ framework for the development of new unit generators for Csound. The original interface for this type work is provided in the C language and it still provides the most complete set of components to cover all possible requirements. CPOF attempts to allow a simpler and more economical approach to creating plugin opcodes. The paper explores the fundamental characteristics of the framework and how it is used in practice. The helper classes that are included in CPOF are presented with examples. Finally, we look at some uses in the Csound source codebase.

1. INTRODUCTION

Plugins in Csound [1] are usually written in C, which provides a low-level access to the engine, allowing an uncompromising and complete scope for new opcode (unit generator) development. For most system developers, this will continue to be the best working environment. However, for many of the more common forms of plugins, this interface can be complex and cumbersome. In particular, we might like to take advantage of an object-oriented approach so that we can, for instance, re-use code more extensively and take advantage of existing algorithms and libraries. For this, the ideal language is probably the C++ in its more modern incarnations [2]. In this paper, we describe a lightweight framework to facilitate programming in this environment, the Csound Plugin Opcode Framework (CPOF – *see-pough* or *cipó*, *vine* in Portuguese). While there is a pre-existing C++ interface for plugin writing in the Csound codebase, this framework provides an alternative to it, that attempts to be thin, simple, complete, and handling internal Csound resources in a safe way (using the same mechanisms provided by the underlying C interface).

1.1 Design Fundamentals

The conditions in which the framework is supposed to work constrain significantly the scope of what is possible. In particular,

1. Csound is written in C. It instantiates opcodes and makes calls to opcode processing functions, but it does not know anything about C++.
2. Virtual functions are a no-go zone (due to 1). Everything needs to be set at compile time.
3. Registering processing functions requires that these are defined as static.
4. Up to three different functions should be registered (for different action times).
5. In C, an opcode dataspace “inherits” from a base structure (`OPDS`) by adding this as its first member variable.

One of the possibilities for designing a framework based on inheritance without virtual functions is to employ a method called *curiously recurring template pattern* (CRTP) [3] . A variation of this is in fact used in the pre-existing C++ opcode development interface. However, we can do better with a much simpler approach. There is no need for a compile-time mimicking of the virtual-function mechanism. This is because it is not our objective to have a general purpose framework for C++ programs, where users would be instantiating their own objects and possibly using generic pointers and references that need to bind to the correct override.

Here there is a much narrower scope: Csound does the instantiation and the calls, so we can make the decision at compile time just by providing functions that hide rather than override (in the virtual-function sense) the base class ones. In this case, hiding plays the same role as overriding. So we can have a plugin base class from which we will inherit to create the actual opcodes. This class will inherit from `OPDS` and provide some extra members that are commonly used by all opcodes. It will also provide stub methods for the processing functions, which then can be ‘specialised’ in the derived classes.

Given that they will not ever be called, it would seem that these stubs are surplus to requirements. However, having these allows a considerable simplification in the plugin registration process. We can just register any plugin in the same way, even if it does only provide one or two of the required processing functions. The stubs play an important role to keep the compiler happy in this scheme, even if Csound will not take any notice of them.

This mechanism requires that we provide function templates for registration. These get instantiated with exactly the derived types and are used to glue the C++ code into

Csound. Each one of them is minimal: consisting just of a call to the instantiated class processing function. For instance, one of these, used at instrument initialisation time, is defined simply as

```
template <typename T>
 int init(CSOUND *csound, T *p) {
  p->csound = (Csound *)csound;
  return p->init();
 }
```

In this case T is derived type passed on registration (which is defined at compile time). When running, Csound calls this function, which in its turn delegates directly to the plugin class in question. Note that this is all hidden from the framework user, who only needs to derive her classes and register them. As we will see in the following sections, this scheme enables significant economy, re-use and reduction in code verbosity (one of the issues with CRTP).

2. THE FRAMEWORK

CPOF is based on two class templates, which plugin classes derive from and instantiate. Opcode classes can then be registered by instantiating and calling one of the two overloaded registration function templates. All CPOF code is declared under the namespace **csnd**. In this section, we discuss the base class templates, three simple cases of derived plugins and their registration.

2.1 The Base Classes

The framework base classes are actually templates which need to be derived and instantiated by the user code. The most general of these is **Plugin**[1]. To use it, we program our own class by subclassing it and passing the number of output and inputs our opcode needs as its template arguments:

```
#include <plugin.h>
struct MyPlug : csnd::Plugin<1,1> { };
```

The above lines will create a plugin opcode with one output (first template argument) and one input (second template argument). This class defines a complete opcode, but since it is only using the base class stubs, it is also fully non-op.

To make it do something, we will need to reimplement one, two or three of its methods. As outlined above, this base class is derived from the Csound structure **OPDS** and has the following members:

- `outargs`: a `Params` object holding output arguments.
- `inargs`: input arguments (`Params`).
- `csound`: a pointer to the Csound engine object.
- `offset`: the starting position of an audio vector (for audio opcodes only).
- `nsmps`: the size of an audio vector (also for audio opcodes only).
- `init()`, `kperf()` and `aperf()` non-op methods, to be reimplemented as needed (see sec. 2.2).

The other base class in CPOF is **FPlugin**, derived from **Plugin**, which adds an extra facility for fsig (streaming frequency-domain) plugins:

- `framecount`: a member to hold a running count of fsig frames.

Input and output arguments, which in C are just raw pointers are conveniently wrapped by the Params class, which provides a number of methods to access them according to their expected Csound variable types.

2.2 Deriving opcode classes

Csound has two basic *action* times for opcodes: init and perf-time. The former runs a processing routine once per instrument instantiation (and/or once again if a re-init is called for). Code for this is placed in the **Plugin** class `init()` function. Perf-time code runs in a loop and is called once every control (k-)cycle. The other class methods `kperf()` and `aperf()` are called in this loop, for control (scalar) and audio (vectorial) processing. The following examples demonstrate the derivation of plugin classes for each one of these opcode types (i, k or a). Note that k and a opcodes can also use i-time functions if they require some sort of initialisation.

2.2.1 Init-time opcodes

For init-time opcodes, all we need to do is provide an implementation of the `init()` method:

```
struct Simplei : csnd::Plugin<1,1> {
  int init() {
    outargs[0] = inargs[0];
    return OK;
  }
};
```

In this simple example, we just copy the input arguments to the output once, at init-time. Each scalar input type can be accessed using array indexing. All numeric argument data is real, declared as **MYFLT**, the internal floating-point type used by Csound.

2.2.2 K-rate opcodes

For opcodes running only at k-rate (no init-time operation), all we need to do is provide an implementation of the `kperf()` method:

```
struct Simplek : csnd::Plugin<1,1> {
  int kperf() {
    outargs[0] = inargs[0];
    return OK;
  }
};
```

Similarly, in this simple example, we just copy the input arguments to the output at each k-period.

[1] All CPOF code is declared in `plugin.h`

2.2.3 A-rate opcodes

For opcodes running only at a-rate (and with no init-time operation), all we need to do is provide an implementation of the `aperf()` method:

```
struct Simplea : csnd::Plugin<1,1> {
 int aperf() {
  std::copy(inargs(0)+offset,
    inargs(0)+nsmps, outargs(0));
    return OK;
  }
};
```

Because audio arguments are `nsmps`-size vectors, we get these using the overloaded **`operator`**`()` for the `inargs` and `outargs` objects, which takes the argument number as input and returns a **`MYFLT`** pointer to the vector.

2.3 Registering opcodes with Csound

Once we have written our opcode classes, we need to tell Csound about them, so that they can be used, for this we use the CPOF function template `plugin()`:

```
template <typename T>
int plugin(Csound *csound,
           const char *name,
           const char *oargs,
           const char *iargs,
           uint32_t thrd,
           uint32_t flags = 0)
```

Its parameters are:

- `csound`: a pointer to the **`Csound`** object to which we want to register our opcode.
- `name`: the opcode name as it will be used in Csound code.
- `oargs`: a string containing the opcode output types, one identifier per argument
- `iargs`: a string containing the opcode input types, one identifier per argument
- `thrd`: a code to tell Csound when the opcode should be active.
- `flags`: multithread flags (generally 0 unless the opcode accesses global resources).

For opcode type identifiers, the most common types are: a (audio), k (control), i (i-time), S (string) and f (fsig). For the thread argument, we have the following options, which depend on the processing methods implemented in our plugin class:

- `thread::i`: indicates `init()`.
- `thread::k`: indicates `kperf()`.
- `thread::ik`: indicates `init()` and `kperf()`.
- `thread::a`: indicates `aperf()`.
- `thread::ia`: indicates `init()` and `aperf()`.
- `thread::ika`: indicates `init()`, `kperf()` and `aperf()`.

We instantiate and call these template functions inside the plugin dynamic library entry-point function `on_load()`. This function needs to implemented only **once** [2] in each opcode library. For example:

```
#include <modload.h>
void csnd::on_load(Csound *csound){
  csnd::plugin<Simplei>(csound,
     "simple", "i", "i",
     csnd::thread::i);
  csnd::plugin<Simplek>(csound,
      "simple", "k", "k",
       csnd::thread::k);
  csnd::plugin<Simplea>(csound,
       "simple", "a", "a",
       csnd::thread::a);
  return 0;
}
```

will register the `simple` *polymorphic* opcode, which can be used with i-, k- and a-rate variables. In each instantiation of the plugin registration template, the class name is passed as an argument to it, followed by the function call. If the class defines two specific static members, `otypes` and `itypes`, to hold the types for output and input arguments, declared as

```
struct MyPlug : csnd::Plugin<1,2> {
  static constexpr
   char const *otypes = "k";
  static constexpr
   char const *itypes = "ki";
  ...
};
```

then we can use a simpler overload of the plugin registration function:

```
template <typename T>
int plugin(Csound *csound,
           const char *name,
           uint32_t thread,
           uint32_t flags = 0)
```

For some classes, this might be a very convenient way to define the argument types. For other cases, where opcode polymorphism might be involved, we might re-use the same class for different argument types, in which case it is not desirable to define these statically in a class.

3. SUPPORT CLASSES

Plugins developed with CPOF can avail of a number of helper classes for accessing resources in the Csound engine. These include a class encapsulating the engine itself,

[2] The header file `modload.h`, where `on_load()` is declared, contains three boilerplate calls to Csound module C functions, required for Csound to load plugins properly. For this reason, each plugin library should also include this header only **once**, otherwise duplicate symbols will cause linking errors.

and classes for resource allocation, input/output access/-manipulation, threads, and support for constructing objects allocated in Csound's heap.

3.1 The Csound Engine Object

Plugins are run by an engine which is encapsulated by the **Csound** class. They all hold a pointer to this, called csound, which is needed for some of the operations invoking parameters, or for some utility methods (such as console messaging, MIDI data access, FFT operations). The following are the public methods of the Csound class:

- init_error(): takes a string message and signals an initialisation error.
- perf_error(): takes a string message, an instrument instance and signals a performance error.
- warning(): warning messages.
- message(): information messages.
- sr(): returns engine sampling rate.
- _0dbfs(): returns max amplitude reference.
- _A4(): returns A4 pitch reference.
- nchnls(): return number of output channels for the engine.
- nchnls_i(): same, for input channel numbers.
- current_time_samples(): current engine time in samples.
- current_time_seconds(): current engine time in seconds.
- midi_channel(): midi channel assigned to this instrument.
- midi_note_num(): midi note number (if the instrument was instantiated with a MIDI NOTE ON).
- midi_note_vel(): same, for velocity.
- midi_chn_aftertouch(), midi_chn_polytouch(),midi_chn_ctl(), midi_chn_pitchbend(): midi data for this channel.
- midi_chn_list(): list of active notes for this channel.
- fft_setup(), rfft(), fft(): FFT operations.

In addition to these, the Csound class also holds a deinit method registration function template for Plugin objects to use:

```
template <typename T>
 void plugin_deinit(T *p);
```

This is only needed if the Plugin has allocated extra resources using mechanisms that require de-allocation. It is not employed in most cases, as we will see below. To use it, the plugin needs to implement a deinit() method and then call the plugin_deinit() method passing itself (through a this pointer) in its own init() function:

```
csound->plugin_deinit(this);
```

3.2 Audio Signals

Audio signal variables are vectors of nsmps samples and we can access them through raw **MYFLT** pointers from input and output parameters. To facilitate a modern C++ approach, the AudioSig class wraps audio signal vectors conveniently, providing iterators and subscript access. Objects are constructed by passing the current plugin pointer (**this**) and the raw parameter pointer, e.g.:

```
 csnd::AudioSig in(this,inargs(0));
 csnd::AudioSig out(this,outargs(0));
 std::copy(in.begin(), in.end(),
           out.begin());
```

3.3 Memory Allocation

For efficiency and to prevent leaks and undefined behaviour we need to leave all memory allocation to Csound and refrain from using C++ dynamic memory allocators or standard library containers that use dynamic allocation behind the scenes (e.g. std::vector).

This requires us to use the AuxAlloc mechanism implemented by Csound for opcodes. To allow for ease of use, CPOF provides a wrapper template class (which is not too dissimilar to std::vector) for us to allocate and use as much memory as we need. This functionality is given by the **AuxMem** class:

- allocate(): allocates memory (if required).
- **operator**[]: array-subscript access to the allocated memory.
- data(): returns a pointer to the data.
- len(): returns the length of the vector.
- begin(), cbegin() and end(), cend(): return iterators to the beginning and end of data.
- iterator and const_iterator: iterator types for this class.

As an example of use, we can implement a simple delay line [4] opcode, whose delay time is set at i-time, providing a slap-back echo effect:

```
struct DelayLine : csnd::Plugin<1,2> {
 csnd::AuxMem<MYFLT> delay;
 csnd::AuxMem<MYFLT>::iterator iter;

 int init() {
  delay.allocate(csound,
      csound->sr()*inargs[1]);
  iter = delay.begin();
  return OK;
 }

 int aperf() {
  csnd::AudioSig in(this, inargs(0));
  csnd::AudioSig out(this, outargs(0));

  std::transform(in.begin(),in.end(),
        out.begin(),[this](MYFLT s){
        MYFLT o = *iter;
        *iter = s;
        if(++iter == delay.end())
                 iter = delay.begin();
         return o;});
  return OK;
 }
};
```

In this example, we use an **AuxMem** interator to access the delay vector. It is equally possible to access each element with an array-style subscript. The memory allocated by this class is managed by Csound, so we do not need to be concerned about disposing of it. To register this opcode, we do

```
csnd::plugin<DelayLine>(csound,
        "delayline", "a", "ai",
        csnd::thread::ia);
```

3.4 Function Table Access

Access to function tables has also been facilitated by a thin wrapper class that allows us to treat it as a vector object. This is provided by the Table class:

- `init()`: initialises a table object from an opcode argument pointer.
- **`operator`**`[]`: array-subscript access to the function table.
- `data()`: returns a pointer to the function table data.
- `len()`: returns the length of the table (excluding guard point).
- `begin()`, `cbegin()` and `end()`, `cend()`: return iterators to the beginning and end of the function table.
- `iterator` and `const_iterator`: iterator types for this class.

An example of table access is given by an oscillator opcode, which is implemented in the following class:

```
struct Oscillator : csnd::Plugin<1,3> {
 csnd::Table tab;
 double scl;
 double x;

 int init() {
  tab.init(csound,inargs(2));
  scl = tab.len()/csound->sr();
  x = 0;
  return OK;
 }

 int aperf() {
  csnd::AudioSig out(this, outargs(0));
  MYFLT amp = inargs[0];
  MYFLT si = inargs[1] * scl;

  for(auto &s : out) {
    s = amp * tab[(uint32_t)x];
    x += si;
    while (x < 0) x += tab.len();
    while (x >= tab.len())
                  x -= tab.len();
  }
  return OK;
 }
};
```

The table is initialised by passing the relevant argument pointer to it (using its `data()` method). Also note that, as we need a precise phase index value, we cannot use iterators in this case (without making it very awkward), so we employ straightforward array subscripting. The opcode is registered by

```
csnd::plugin<Oscillator>(csound,
        "oscillator", "a", "kki",
        csnd::thread::ia);
```

3.5 String Types

String variables in Csound are held in a `STRINGDAT` data structure, containing a data member that holds the actual string and a size member with the allocated memory size. While CPOF does not wrap strings, it provides a translated access to string arguments through the argument objects `str_data()` function. This takes an argument index (similarly to `data()`) and returns a reference to the string variable, as demonstrated in this example:

```
struct Tprint : csnd::Plugin<0,1> {
 int init() {
  char *s = inargs.str_data(0).data;
  csound->message(s);
  return OK;
 }
};
```

This opcode will print the string to the console. Note that we have no output arguments, so we set the first template parameter to 0. We register it using

```
csnd::plugin<Tprint>(csound, "tprint",
        "", "S",  csnd::thread::i);
```

3.6 Streaming Spectral Types

For streaming spectral processing opcodes, we have a different base class with extra facilities needed for their operation (**FPlugin**). In Csound, fsig variables, which carry spectral data streams, are held in a **PVSDAT** data structure.

To facilitate their manipulation, CPOF provides the **Fsig** class, derived from **PVSDAT**. To access phase vocoder bins, a container interface is provided by **pv_frame** (**spv_frame** for the sliding mode)[3]. This holds a series of **pv_bin** (**spv_bin** for sliding)[4] objects, which have the following methods:

- amp(): returns the bin amplitude.
- freq(): returns the bin frequency.
- amp(**float** a): sets the bin amplitude to a.
- freq(**float** f): sets the bin frequency to f.
- **operator***(**pv_bin** f): multiply the amp of a pvs bin by f.amp.
- **operator***(**MYFLT** f): multiply the bin amp by f
- **operator***=(): unary versions of the above.

The **pv_bin** class can also be translated into a std::complex<**float**>, object if needed. This class is also fully compatible with the C complex type and an object obj can be cast into a float array consisting of two items (or a float pointer), using **reinterpret_cast**<**float**(&)[2]>(obj) or **reinterpret_cast**<float*>(&obj). The Fsig class has the following methods:

- init(): initialisation from individual parameters or from an existing fsig. Also allocates frame memory as needed.
- dft_size(),hop_size(),win_size(),win_type() and nbins(), returning the PV data parameters.
- count(): get and set fsig framecount.
- isSliding(): checks for sliding mode.
- **fsig_format**(): returns the fsig data format (**fsig_format**::pvs, ::polar ::complex, or ::tracks).

The **pv_frame** (or **spv_frame**) class contains the following methods:

- **operator**[]: array-subscript access to the spectral frame
- data(): returns a pointer to the spectral frame data.
- len(): returns the length of the frame.
- begin(), cbegin() and end(), cend(): return iterators to the beginning and end of the data frame.
- iterator and const_iterator: iterator types for this class.

Fsig opcodes run at k-rate but will internally use an update rate based on the analysis hopsize. For this to work, a frame count is kept and checked to make sure we only process the input when new data is available. The following example class implements a simple gain scaler for fsigs:

```
struct PVGain : csnd::FPlugin<1, 2> {
 static constexpr
        char const *otypes = "f";
 static constexpr
         char const *itypes = "fk";

 int init() {
  if(inargs.fsig_data(0).isSliding()){
   char *s = "sliding not supported";
   return csound->init_error(s);
  }
  if(inargs.fsig_data(0).fsig_format()
     != csnd::fsig_format::pvs &&
     inargs.fsig_data(0).fsig_format()
     != csnd::fsig_format::polar){
     char *s = "format not supported";
     return csound->init_error(s);
  }
  csnd::Fsig &fout =
        outargs.fsig_data(0);
  fout.init(csound,
        inargs.fsig_data(0));
  framecount = 0;
  return OK;
 }

 int kperf() {
   csnd::pv_frame &fin =
        inargs.fsig_data(0);
   csnd::pv_frame &fout =
        outargs.fsig_data(0);
   uint32_t i;

   if(framecount < fin.count()) {
     std::transform(fin.begin(),
       fin.end(), fout.begin(),
       [this](csnd::pv_bin f){
        return f *= inargs[1];});
     framecount =
        fout.count(fin.count());
    }
    return OK;
  }
};
```

Note that, as with strings, there is a dedicated method in the arguments object that returns a ref to an Fsig class (which can also be assigned to a **pv_frame** ref). This

[3] **pv_frame** is a convenience typedef for **Pvframe**<**pv_bin**>, whereas **spv_frame** is **Pvframe**<**spv_bin**>

[4] **pv_bin** is **Pvbin**<**float**> and **spv_bin** is **Pvbin**<**MYFLT**>

is used to initialise the output object at i-time and then to obtain the input and output variable data Csound processing. The framecount member is provided by the base class, as well as the format check methods. This opcode is registered using

```
csnd::plugin<PVGain>(csound, "pvg",
        csnd::thread::ik);
```

3.7 Array Variables

Opcodes with array inputs or outputs use the data structure **ARRAYDAT** for parameters. Again, in order to facilitate access to these argument types, CPOF provides a wrapper class. The framework currently supports only one-dimensional arrays directly. The template container class **Vector**, derived from ARRAYDAT, holds the argument data. It has the following members:

- init(): initialises an output variable.
- **operator**[]: array-subscript access to the vector data.
- data(): returns a pointer to the vector data.
- len(): returns the length of the vector.
- begin(), cbegin() and end(), cend(): return iterators to the beginning and end of the vector.
- iterator and const_iterator: iterator types for this class.
- data_array(): returns a pointer to the vector data.

In addition to this, the inargs and outargs objects in the **Plugin** class have a template method that can be used to get a **Vector** class reference. A trivial example is shown below:

```
struct SimpleArray : csnd::Plugin<1, 1>{
  int init() {
    csnd::Vector<MYFLT> &out =
        outargs.vector_data<MYFLT>(0);
    csnd::Vector<MYFLT> &in =
        inargs.vector_data<MYFLT>(0);
    out.init(csound, in.len());
    return OK;
  }

  int kperf() {
    csnd::Vector<MYFLT> &out =
        outargs.vector_data<MYFLT>(0);
    csnd::Vector<MYFLT> &in =
        inargs.vector_data<MYFLT>(0);
    std::copy(in.begin(), in.end(),
        out.begin());
    return OK;
  }
  };
```

This opcode is registered using the following line:

```
csnd::plugin<SimpleArray>(csound,
        "simple", "k[]", "k[]",
        csnd::thread::ik);
```

3.8 Multithreading

CPOF supports the creation of threads through the **Thread** pure virtual class. Developers wanting to avail of a subthread for processing can derive their own thread class from this and implement its run() method.

3.9 Constructing member variables

Plugin classes can, in general, be composed of member variables of any type, built in or user defined. However, we have to remember that opcodes are allocated and instantiated by C code, which does not know anything about classes. A member variable of a non-trivial class will not be constructed at instantiation. This is perfectly fine for all CPOF classes, which are designed to expect this. However, this might cause problems for other classes that are external to the framework. In this case, a placement **new** needs to be employed at init time to construct an object declared as a plugin class member (and thus allocated in Csound's heap). To facilitate matters, CPOF includes a template function that can be used to construct any member objects:

```
template <typename T,
   typename ... Types>
   T *constr(T* p, Types ... args){
      return new(p) T(args ...);
   }
```

For instance, let's say we have in our plugin an object of type A called obj. To construct this, we just place the following line in the plugin init() method:

```
csnd::constr(&obj,10,10.f);
```

where the arguments are the variable address, followed by any class constructor parameters. Note that if the class allocates any resources, we will need to invoke its destructor explicitly through **csnd**::destr(&obj) in a deinit() method.

4. BUILDING THE OPCODE LIBRARY

To build a plugin opcode library, we require a C++ compiler supporting the C++11 [5] standard (-std=c++11), and the Csound public headers. CPOF has no link dependencies (not even to the Csound library). The opcodes should be built as a dynamic/shared module (e.g .so on Linux, .dylib on OSX or .dll on Windows).

5. EXAMPLES FROM THE CSOUND SOURCES

CPOF is already being used in the development of new opcodes for Csound. It allows very compact and economical code, especially in conjunction with some of the more modern facilities of C++. For example, the following class template is used to generate a whole family of numeric array-variable operators for i-time and k-rate processing [5]:

```
template<MYFLT (*op)(MYFLT)>
struct ArrayOp : csnd::Plugin<1, 1>{
 int oprt(csnd::myfltvec &out,
         csnd::myfltvec &in){
    std::transform(in.begin(),in.end(),
           out.begin(), [](MYFLT f){
                return op(f);});
    return OK;
  }
 int init() {
    csnd::myfltvec &out =
     outargs.myfltvec_data(0);
    csnd::myfltvec &in =
     inargs.myfltvec_data(0);
    out.init(csound,in.len());
    return oprt(out, in);
  }
 int kperf() {
  return oprt(outargs.myfltvec_data(0),
        inargs.myfltvec_data(0));
  }
};
```

A plugin implementing i-time $\cos(x)$ where x is an array is created with the following line:

```
csnd::plugin<ArrayOp<std::cos>>(csound,
 "cos", "i[]", "i[]",csnd::thread::i);
```

whereas for a k-rate $\exp(x)$ is implemented by:

```
csnd::plugin<ArrayOp<std::exp>>(csound,
 "exp", "k[]", "k[]",csnd::thread::ik);
```

Forty-six such operators are currently implemented reusing the same code. Another twelve use a similar class template for binary operations (2 inputs).

Another example shows the use of standard algorithms in spectral processing. The following new opcode implements *spectral tracing* [6], which retains only a given number of bins in each frame, according to their amplitude. To select the bins, we need to sort them to find out the ones we want to retain (the loudest N). For this we collect all amplitudes from the frame and then apply *nth element* sorting, placing the threshold amplitude in element n. Then we just filter the original frame according to this threshold. Here we have the performance code (`amps` is a dynamically allocated array belonging to the **`Plugin`** object).

```
int kperf() {
 csnd::pv_frame &fin =
      inargs.fsig_data(0);
 csnd::pv_frame &fout =
      outargs.fsig_data(0);

 if(framecount < fin.count()) {
  int n =  fin.len()-(int)inargs[1];
  float thrsh;
  std::transform(fin.begin(),
   fin.end(), amps.begin(),
   [](csnd::pv_bin f){
         return f.amp();});

  std::nth_element(amps.begin(),
         amps.begin()+n, amps.end());
  thrsh = amps[n];

  std::transform(fin.begin(),
   fin.end(),fout.begin(),
   [thrsh](csnd::pv_bin f){
          return f.amp() >= thrsh ?
           f : csnd::pv_bin(); });
   framecount =
    fout.count(fin.count());
 }
 return OK;
}
```

6. CONCLUSIONS

This paper described CPOF and its fundamental characteristics. We looked at how the base classes are constructed, how to derive from them, and register new opcodes in the system. The framework is designed to support modern C++ idioms and adopts the C++11 standard. All of the code examples discussed in this paper are provided in `opcodes.cpp`, found in the `examples/plugin` directory of the Csound source codebase [6]. CPOF is part of Csound and is distributed alongside its public headers. Csound is free software, licensed by the Lesser GNU Public License.

7. REFERENCES

[1] V. Lazzarini, J. ffitch, S. Yi, J. Heintz, Ø. Brandtsegg, and I. McCurdy, *Csound: A Sound and Music Computing System.* Springer Verlag, 2016.

[2] B. Stroustrup, *The C++ Programming Language*, 4th ed. Addison-Wesley, 2013.

[3] D. Abrahams and A. Gurtovoy, *C++ Template Metaprogramming: Concepts, Tools, and Techniques from Boost and Beyond (C++ in Depth Series).* Addison-Wesley Professional, 2004.

[4] V. Lazzarini, "Time-domain signal processing," in *The Audio Programming Book*, R. Boulanger and V. Lazzarini, Eds. MIT Press, 2010, pp. 463–512.

[5] ISO/IEC, "ISO international standard ISO/IEC 4882:2011, programming language C++," 2011. [Online]. Available: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=50372

[6] T. Wishart, *Audible Design.* Orpheus The Pantomine, 1996.

[5] The convenience typedef **`myfltvec`** for **`Vector<MYFLT>`** is employed here

[6] `https://github.com/csound/csound`

Versioning and Annotation Support for Collaborative Mapping Design

Johnty Wang
Input Devices and Music
Interaction Lab/CIRMMT
McGill University
Johnty.Wang@mcgill.ca

Joseph Malloch
Graphics and Experiential Media Lab
Dalhousie University
Joseph.Malloch@dal.ca

Stéphane Huot
Mjolnir Group
Inria Lille
Stephane.Huot@inria.fr

Fanny Chevalier
Mjolnir Group
Inria Lille
Fanny.Chevalier@inria.fr

Marcelo Wanderley
Input Devices and Music
Interaction Lab/CIRMMT
McGill University
Marcelo.Wanderley@mcgill.ca

ABSTRACT

This paper describes a versioning and annotation system for supporting collaborative, iterative design of mapping layers for digital musical instruments (DMIs). First we describe prior experiences and contexts of working on DMIs that has motivated such features in a tool, describe the current prototype implementation and then discuss future work and features that are intended to improve the capabilities of tools for new musical instrument building as well as general interactive applications that involve the design of mappings with a visual interface.

1. INTRODUCTION

A crucial component of the digital musical instrument (DMI) [1] building process involves the mapping of sensor or gesture input signals from the musician to relevant synthesis parameters, and is described in detail by [2]. While there are many different approaches to the design and implementation of DMI mapping, for many designers, a common part of the process involves the use of graphical user interfaces where the connection of signals can be observed and manipulated visually. In this paper we use personal experiences of building DMIs in a variety of contexts to motivate two main features for a mapping tool: a deeply integrated graphical versioning and comparison tool, and rich annotation features that go beyond the text-based tagging commonly employed by existing versioning systems. While our focus here is to extend the capabilities of tools for creative design of interactive systems [3], we acknowledge the fact that a good tool is not the end-all solution in itself, but something that can help in the process. In this paper we first present background and related work, followed by the motivations for and details on the implementation of our system, and finally a discussion on the limitations of the current work and future steps.

2. EXPERIMENTATION AND ITERATION

Over the years we have been involved in the conception, design, use and evaluation of a large number of DMIs, including some that have been actively performed for 10 years and have toured internationally. Throughout the development of these instruments, *iteration* has proven to be the most important aspect. Many designers take a holistic approach their DMI concept: designing not just the sensing system or sound synthesizer but also gesture vocabularies, notation systems, and lighting or video systems that complement their idea of performance with the instrument. This approach is exciting, but also brings with it difficult challenges, since the various aspects of the instrument are interdependent - the physical instrument, sensing system, gesture, and sound cannot be changed individually without affecting the others. While the overall concept of the instrument can inform process, the actual implementation often requires iteration. Usually, the most important advances are due to a complex system *surprising* the designer, inspiring them to modify their vision of the final instrument.

It is important that our tools for supporting mapping design support iterative workflows, and also make it easy to do “exploratory” development which might result in serendipitous discovery of new directions for the instrument.

2.1 Collaboration

If iteration is important in solo development, in collaborative design contexts it is essential, especially in groups where the roles of hardware designer, composer, and performer are played by different collaborators. In this case it is no longer feasible for a single united vision of the instrument (and its performance practice and repertoire) to exist. The collaborators inevitably use different tools, have different approaches, and in interdisciplinary projects even different vocabularies for discussing interaction. During the course of several long-term projects [4], we started building approaches and tools to try to support interdisciplinary collaborations based on a few principles:

- **Don't enforce representation standards:** Encouraging each collaborator to represent their system component in the strongest way possible helps maintain the benefits of their specific domain-knowledge to the project. It also helps with modularity!

- **Make things freely connectable:** If we encourage strong but idiomatic representations of system components, flexibility must come from the ability to freely interconnect them. Our approach is for *models* to "live" only in devices rather than influencing connection or communication protocols; representations *on the wire* should be low-level, allowing basic compatibility between drastically different devices.

- **Don't dumb it down, but...:** Musicians possess highly technical knowledge, and are usually capable of learning surprising amounts of electrical engineering, digital signal processing, and software design in short periods of time. This means that instrument designers do not need to hide low-level technical properties of the system from the performer, however the nature of the system representation should be questioned: is there a technical representation from the performer's perspective that might serve better?

The software library *libmapper* [4] and the surrounding tools are the result of this design philosophy and have supported a large number of interdisciplinary research/creation projects. These tools form a discoverable, distributed network of mappable devices; they encourage (but do not enforce) modular, strong representations of system components; they glue together different programming languages and environments; and they aid the process of rapid experimentation and iteration by making every signal compatible and allowing drag-and-drop style mapping design.

3. RELATED WORK

There are a fairly large number of existing tools for supporting the task of mapping DMIs or similar interactive systems. Some, like the Mapping Library for Pure Data [5] and the Digital Orchestra Toolbox [1] try to provide building blocks for assembling mapping structures; others, like MnM [6] and LoM [7] support specific multidimensional mapping schemes, usually learned from examples. A variety of platforms and environments are also used for designing and implementing DMI mapping, including ICon [8], junXion [9], and Jamoma [10]. Finally, some commercial DMIs have dedicated instrument-specific mapping systems, such as the Continuum Editor [11], 2PIM [2] , Karlax Bridge [3] , and EigenD [4]

3.1 Version Control Systems

The core features of this work is related to two particular aspects of version control: first, the comparison between two different versions of a document [12] and second, the management of previous versions. While the workflow of most modern source control systems are centred around these two issues, they mostly focus on text based data. When the data has a graphical view, or is based on a translation of text data, a text-based "diff" tool is no longer sufficient. Dragicevic et al. [13] describes the process of implementing a comparison system via computing the visual output and then rendering the transition between versions using animations, which preserves the mental model of the user better when there is a translation process is involved.

The environment implemented by Rekall [14], designed to serve both as a "score" as well as documentation for a theatrical work, contains features that are relevant to our application since it contains a visual timeline as well as a rich annotation interface, but is intended for a very different purpose of documenting the timeline progression of a piece of work, as opposed to being able to preview/revert to prior states.

4. SUPPORTING EXPLORATION, BUILDING COMPLEXITY

4.1 Problems Identified

Through the process of working with DMIs in a variety of contexts as described earlier, we have identified the following issues when it comes to the design of mappings:

- **Fear of losing current state:** By not keeping a constant stack of actions and having to manually manage incremental versions, the fear of moving away from a desirable configuration when experimenting inhibits exploration.

- **Difficulty indexing, saving, finding, restoring and talking** about previous mapping designs limits number of options that can be reasonably compared.

- **Support for complexity:** Without effective comparison between mapping configurations, it is difficult to create complex mappings by combining mapping vignettes generated in workshop settings.

4.2 Proposed Solutions

Based on the issues identified above, we have proposed the following solutions:

- ***Distributed Components*:** Using the software library *libmapper* [4] enables simple composition of new or existing system components, and its distributed nature supports collaboration (For example, an arbitrary number of user interfaces can be actively viewing and editing the system at the same time, with different visual representations of the system).

- **Version tracking:** Ability to store previous states as to encourage experimentation while ensuring that no valuable work is lost.

[1] `http://idmil.org/software/dot`
[2] `http://www.pucemuse.com/`
[3] `http://www.dafact.com/`
[4] `http://www.eigenlabs.com/`

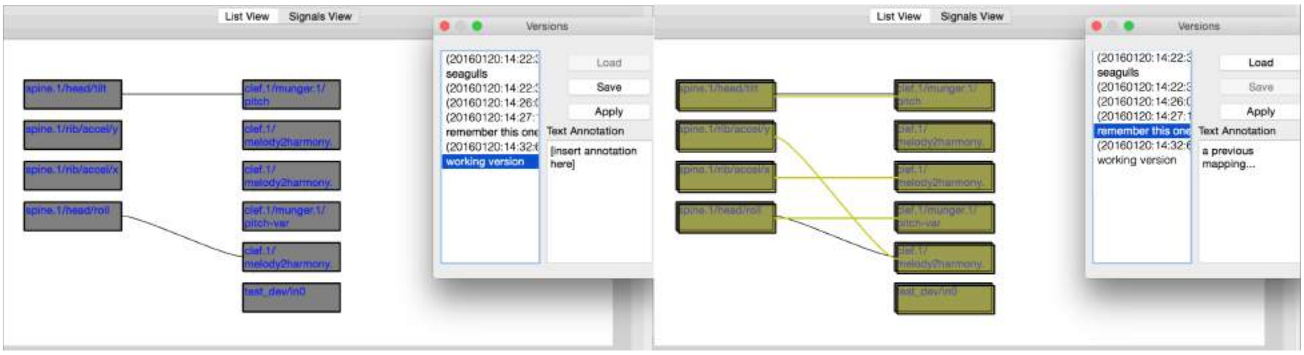


Figure 1. Screenshot showing working version (left), and a comparison of a previous version (right) when it is selected from the interface showing. Note: the actual rendering of the preview overlay has been offset in the application to show up better on a printed page in grayscale.

- **Tagging:** For identifying previous versions. Can be basic text annotations, but also multimedia content that may reflect the creative process, or even recordings of the output of the mapping.
- **Graphical tools:** To realize the above features of tagging, looking at and searching through versioning history, loading and previewing previous versions and creating new ones.

The following section describes the first software application we have implemented to realize the above.

5. IMPLEMENTATION

We have built a graphical tool for *libmapper* that manages distributed mappings between signals. The C++ QT Framework was chosen for its widespread usage, cross platform support, user interface and data model libraries as well as easy interface with the *libmapper* codebase that was written in C/C++.

5.1 Software Features

The mapping tool has the following main features:

- A *libmapper* interface that allows querying of available devices and signals on the network, and interactively modify the mapping configurations
- A user interface that displays the device signals and allows the user to create and break mappings between output and input signals
- A user interface for displaying and previewing previously stored versions of the mapping, along with annotations as well as saving new versions of the mapping

Figure 1 shows screenshots of the application in use and displays two states of operation. The left shows the current "working version" of the mapping, and the right when a previous version is being previewed. Since the *libmapper* network is a distributed system, a third layer, showing the actual status of the live network, can be overlaid on top and synchronized with the working version (not currently shown in the screenshot).

6. FUTURE WORK

In addition to continuous development in refining the implementation described above, there are a few additional features we are currently considering which would increase the capabilities of the tool, and provide interesting possibilities.

- **Constant storage of state changes**: In addition to explicitly stored states, a stack of all user input could be stored into the versioning system which essentially creating an undo-redo loop that is always stored. Versioning system backends such as **.git**[5] provides efficient ways of handling this kind of incremental changes in data.
- **More featured tagging with data snippets, other media**: by recording segments of input signal streams and associating them with particular versions, it would be possible to "replay" the input at a given point in time. This also allows alternative mappings to be applied after the fact, which allows experimentation to occur without having direct access to live inputs, which may be a limited resource in collaborative projects involving many individuals. Using media tags allow more expressive annotation options.
- **Advanced Query features**: by applying analysis on the input stream and classifying/segmenting relevant gestures and activities, a prior version could be queried using gestural input. Also, queries using recordings of the output can also be very useful in a collaborative setting as it does not require in depth technical knowledge of the mapping itself.
- **Additional visualizations of the mapping**: Using different data visualization techniques to visually represent the connections between signals may provide additional insights not seen by one particular view. This also may include revealing changes in the parameter processing and transformation features of libmapper, in addition to signal connections themselves.

[5] `http://www.git-scm.com/`

7. CONCLUSION

We have discussed the motivation for integrating a versioning control and annotation system for visual mapping tools for DMI design based on prior experience working in a variety of contexts. Being able to effectively store, preview, and reload previous configurations of a system addresses some of the challenges of this creative process by reducing risks, encouraging exploration, and improving the management of configuration files. A newly developed mapping tool for *libmapper*, a well used system for connecting signals employed for the design and implementation of DMIs, is implemented with these features and described. These solutions, while motivated by our own experience, also correspond to the findings by others in the general case [3] for "rethinking user interfaces" for general creativity support tools in general as well as which includes *history keeping, exploratory search, visualization, collaboration and composition*. Such views are also echoed when exploring the development of tools to support computer music [15].

Acknowledgments

The primary author is supported by a NSERC-FRQNT Industrial Innovation Scholarship with Infusion Systems, and the Centre for Interdisciplinary Research in Music Media and Technology.

8. REFERENCES

[1] E. R. Miranda and M. M. Wanderley, *New digital musical instruments: control and interaction beyond the keyboard*. AR Editions, Inc., 2006, vol. 21.

[2] A. Hunt, M. M. Wanderley, and M. Paradis, "The importance of parameter mapping in electronic instrument design," *Journal of New Music Research*, vol. 32, no. 4, pp. 429–440, 2003.

[3] B. Shneiderman *et al.*, "Creativity support tools: Report from a U.S. national science foundation sponsored workshop," *International Journal of Human-Computer Interaction*, vol. 20, no. 2, pp. 61–77, 2006.

[4] J. Malloch, S. Sinclair, and M. M. Wanderley, "Distributed tools for interactive design of heterogeneous signal networks," *Multimedia Tools and Applications*, vol. 74, no. 15, pp. 5683–5707, 2014.

[5] H. C. Steiner, "Towards a catalog and software library of mapping methods," in *Proceedings of the 2006 conference on New interfaces for musical expression*. IRCAM—Centre Pompidou, 2006, pp. 106–109.

[6] F. Bevilacqua, R. Müller, and N. Schnell, "Mnm: a max/msp mapping toolbox," in *Proceedings of the 2005 conference on New interfaces for musical expression*, 2005, pp. 85–88.

[7] D. Van Nort and M. M. Wanderley, "The LoM mapping toolbox for Max/MSP/Jitter," in *Proceedings of the International Computer Music Conference*, New Orleans, USA, 2006.

[8] P. Dragicevic and J.-D. Fekete, "The input configurator toolkit: towards high input adaptability in interactive applications," in *Proceedings of the working conference on Advanced visual interfaces*. ACM, 2004, pp. 244–247.

[9] Steim Foundation, "junXion v4 manual," Online, 2008, available: `http://www.steim.org/steim/support/manuals/jX` `Manual.pdf`. Accessed October 16, 2009.

[10] T. Place and T. Lossius, "Jamoma: A modular standard for structuring patches in max," in *Proceedings of the International Computer Music Conference*, 2006, pp. 143–146.

[11] E. Eagen and L. Haken, *EagenMatrix User Guide*, 2013.

[12] J. W. Hunt and D. M. Mcllroy, "An algorithm for differential file comparison," Bell Laboratories, Tech. Rep., 1976.

[13] P. Dragicevic, S. Huot, and F. Chevalier, "Gliimpse: Animating from markup code to rendered documents and vice versa," in *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, ser. UIST '11. New York, NY, USA: ACM, 2011, pp. 257–262.

[14] C. Bardiot, "Rekall," *Performance Research*, vol. 20, no. 6, pp. 82–86, 2015.

[15] R. A. Fiebrink, D. Trueman, C. Britt, M. Nagai, K. Kaczmarek, M. Early, A. Hege, and P. R. Cook, "Toward understanding human-computer interaction in composing the instrument," in *Proceedings of the International Computer Music Conference*, 2010, pp. 135–142.

Tools for Annotating Musical Measures in Digital Music Editions

Yevgen Mexin, Aristotelis Hadjakos, Axel Berndt, Simon Waloschek
Center of Music and Film Informatics
University of Music Detmold, Germany
OWL University of Applied Sciences, Germany
{e.mexin,a.hadjakos,a.berndt,s.waloschek}@cemfi.de

Anastasia Wawilow, Gerd Szwillus
Human-Computer Interaction and Software Technology Group
Paderborn University, Germany
{apetkau,szwillus}@upb.de

ABSTRACT

The process of creating historical-critical digital music editions involves the annotation of musical measures in the source materials (e.g. autographs, manuscripts or prints). This serves to chart the sources and create concordances between them. So far, this laborious task is barely supported by software tools. We address this shortcoming with two interface approaches that follow different functional and interaction concepts. *Measure Editor* is a web application that complies with the WIMP paradigm and puts the focus on detailed, polygonal editing of image zones. *Vertaktoid* is a multi-touch and pen interface where the focus was on quick and easy measure annotation. Both tools were evaluated with music editors giving us valuable clues to identify the best aspects of both approaches and motivate future development.

1. INTRODUCTION

Musicologists create historical-critical editions of musical works based on autographs by the composer and other source materials, e.g., letters or materials from various copyists and publishers. In order to create a musical score that is both accurate and usable by musicians, the editor has to spot and analyze differences between the different sources, decide which variant to use in the final score, and document the editorial decisions. These decisions are usually described in the critical apparatus, which may be contained in the same volume as the musical score or be published separately. A musician can consult the critical apparatus in order to understand editorial decisions, to examine differences between different editions of the same work, or to spot interesting variants that he or she might choose to play in opposition to the editor's choices.

Digital and hybrid editions, which are published both in digital and printed form, make it easier for a musician to delve into the original materials and understand its inherent ambiguities. On the one hand digital editions are not limited by printing costs and can provide scans of the original autographs, on the other hand suitable human-computer interaction makes it easier to navigate in the complex network of cross-references between various source materials and the editors annotations.

The hybrid edition of Max Reger's works may serve as an example. Two screenshots are depicted in Figure 1. On the left side, the typeset final score is shown together with small icons that represent annotations by the editor. When the user opens an annotation, a window shows the textual annotation as well as the source materials that the annotation refers to (see the lower-left sub-window of the right screenshot in Figure 1). The textual annotation in the depicted example says that one of the sources lacks the fortissimo in measure 3. By clicking on the autograph icons, the corresponding autograph scans are displayed (see the other three sub-windows of the right screenshot). This way, a musician can delve into the piece and understand its heritage and its ambiguities.

Music in common music notation is structured in musical *measures*. Larger works can additionally be split up in several *movements*. Musicians usually refer to specific positions within a score by measure numbers. Measure numbers therefore provide a reasonable granularity for links between textual annotations and the original material. In practice, this makes it necessary to define processable markings for every measure in all related autographs and create concordances, i.e., one-to-one mappings, between these annotation. The XML-based MEI (Music Encoding Initiative) [2] [3] data format, the main format for digital music editions, is able to handle measure definitions in images in the form of polygons. Since musical works can consists of hundreds or even thousands of measures, the annotation of them can, however, be a tedious task. Therefore specialized and time-efficient tools are needed.

We developed two different interfaces: The *Measure Editor* is a web application and part of a full featured music edition platform in development. *Vertaktoid* is a standalone Android Application that supports pen and touch interaction. Both software prototypes were evaluated by music

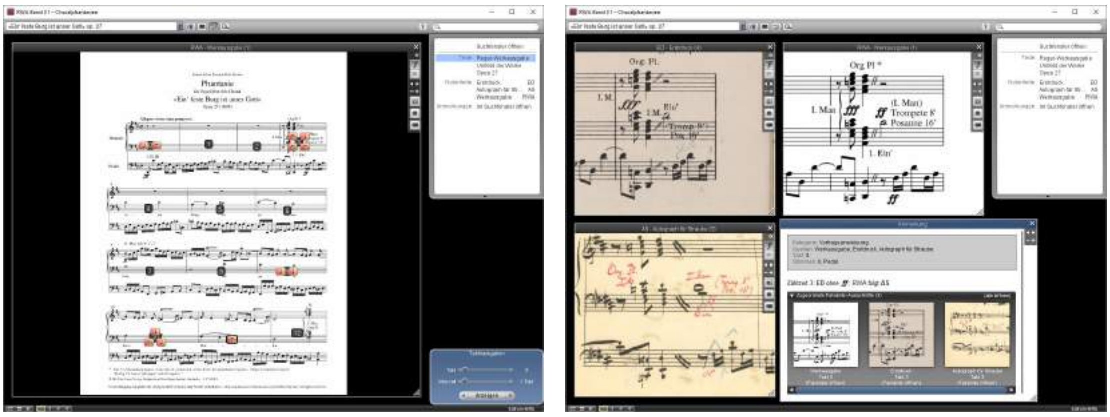


Figure 1. Screenshots from the digital Reger edition [1]

editors, showing potential paths towards an efficient measure annotation tool.

2. RELATED WORK

2.1 Current tools for Measure Annotation

Edirom [1] is a project that aims to assist the music editors providing special software. Two of them are explicitly relevant for this overview: Edirom-Editor [4] and Edirom-Online [5].

Edirom-Editor is a platform that combines different tools for creating digital and hybrid music editions. Edirom-Editor supports the following tasks:

- Cataloging of music works represented by multiple sources with musical and eventual textual content.
- Definition of the music score's structure by specifying the movements and their names.
- Measure annotation with sequence numbers and eventual additional meta-information. The measures will be also aligned to the movements in this step.
- Creation of concordances for the movements and measures that represent the relations between the corresponding elements from different sources.
- Creation of textual annotations and additional references between the elements.
- Export of the created digital music edition in the MEI format.

The MEI format can then be presented and published using Edirom-Online. Edirom-Online can present the music score, the corresponding measures from different sources, the annotations and references. The publishing can be performed both locally and remotely via the web.

Edirom-Editor is well known by music editors, who work on digital and hybrid music editions. Among other projects, Edirom-Editor is being used in the following projects: Detmolder Hoftheater [6], OPERA, [2] and Beethovens Werkstatt [7].

Since measure annotation is the most laborious repetitive task, we developed specialized tools. The development of these tools addressed shortcomings of Edirom-Editor with regard to its usability. In this we see the biggest potential for accelerating the process of measure annotation.

2.2 Measure Detection in Optical Music Recognition

Annotation of measures is a relatively time-consuming process that needs a lot of user attention. In this paper we discuss two tools that facilitate this task by introducing improved interaction techniques and intelligent approaches like automatic numbering (see Section 3.2.2), but the main part of the annotation process remains manual. On one hand, this allows the working process to be free from issues that can be caused by automatic recognition and allows non-standard context-based user decisions. On the other hand, automatic processing of big volume of data counts can be performed much faster.

Optical Music Recognition (OMR) [8] is the automatic conversion of scanned music scores into computer readable data in variable formats, e.g., MusicXML, [3] or MEI. [4] OMR processes the content of music scores, trying to recognize the notation and create a representation that provides the best correspondence to the original source, which is typically scanned. The effectiveness of optical recognition of printed music scores is greater than of hand-written scores, which are very common material for digital music editions. For our purposes the recognition of music notes themselves, including pitch and duration, is not relevant since we are mainly interested in finding measure boundaries. Therefore we will present related work that tackles this task below.

Dalitz and Crawford have investigated the possibility of OMR in relation to the lute tablature. The recognition of measures is discussed in detail in the paper [9]. Although the lute tablature does not rely to common Western music notation, it has a similar measure concept. The staff lines are removed during the preprocessing step and then the algorithm tries to detect the bar line candidates through their *aspect ratio* (ratio of width to height). The candidates can be also a combination of multiple closely located frag-

[1] `http://edirom.de`, last access: Feb. 2017
[2] `http://opera.adwmainz.de`, last access: Feb. 2017
[3] `https://musicxml.com`, last access: Feb. 2017
[4] `http://music-encoding.org`, last access: Feb. 2017

ments. The properties of bar line candidates are validated trough comparison with staff line and staff space parameters. The optical measure recognition for lute tablatures can be performed relatively robustly because they do not contain complex systems of multiple voices such as in other music score.

Vigliensoni, Burlet and Fujinaga have devised an approach for optical measure recognition of music scores in common Western music notation [10]. Vigliensoni et al. follow the task model from Bainbridge and Bell [11] that contains the following steps: image preprocessing and normalization, staff line identification and removal, musical object location and musical reasoning.

The approach by Vigliensoni et al. [10] needs additional information from the user about the structure of staves in music score. This information consists of the count of staves for each page, the count of systems, the relation between staves and systems and the kind of bar lines. During the process of bar line detection, the algorithm has to remove all horizontal lines and filter the resulting set to include only thin, vertical elements. The remaining bar candidates are further filtered by their *aspect ratio* using a user-defined parameter. The algorithm can handle the broken candidates by combining the vertical lines with the nearest horizontal positions. As a final step, the height and position of each bar candidate is compared with the related properties of the corresponding system. The candidates that do not match the system will be also removed. The second user-defined parameter, *vertical tolerance*, controls the sensitivity of the last step. This measure recognition technique was evaluated trough comparison with manually annotated music scores. Vigliensoni et al. have chosen one hundred music score pages from the International Music Score Library Project (IMSLP) [5] and had these pages marked by experienced annotators. The authors compared the these bounding boxes with those recognized by the machine. The results of machine recognition was considered correct if the divergence of the bounding boxes was less than 0.64 cm. The evaluation results yield to an average f-score of 0.91 by the best chosen *aspect ratio* and *vertical tolerance* parameters.

Although a lot of further research papers in OMR thematic field exist, we focus on these two articles because they have contributed largely to optical measure recognition. The approach from Padilla et al. [12] with the clever idea to compare and combine the OMR outputs for different sources of the same music score can be also mentioned. The majority voting approach is used to decide which elements are correctly recognized.

2.3 Interaction

The process of measure annotation comprises three interaction tasks: navigation through autographs, definition of zones (measures, respectively) in the autographs, and editing of zone numbering and movement association. The latter is conveniently solved via text input. Regarding navigation between and within images (autograph pages), standard gestures for panning and zooming are well established and implemented in every multitouch enabled operating system. These will be applied here, too. The definition of zones (typically bounding boxes) is the dominating task. It is essentially similar to the more general task of selecting image regions which is well-researched in the field of human-computer interaction. In Edirom Editor [4] zones are defined by the traditional rectangle spanning gesture that is known, e.g., from graphics editors and file explorers (selecting multiple files by spanning a box around them). Edirom Editor relies on mouse input. Users describe this interaction technique as slow and demanding, requiring a lot of corrections as the initially drawn bounding boxes rarely embrace their content perfectly. The original developers of Edirom Editor obtained this feedback from their own experiences with music edition projects, active communication with other editors and during the yearly Edirom Summer School. This was the situation that motivated the developments which we report here.

As an alternative to mouse input, pen (and touch) seems the most promising modality as it corresponds more closely to the way we naturally work with sheet music. However, there is a variety of selection gestures for pen input, each with its advantages and disadvantages depending on the application context. For some editions it may suffice when zones roughly embrace measures, others may need very precise bounding volumes.

Typical selection gestures in the literature are *spanning a rectangle* (as implemented in Edirom Editor), *lassoing* (well-known from graphics editors) and *tapping* (discrete object selection) [13]. These are sometimes combined with a *mode button* that is operated by the secondary hand to clarify conflicting gestures, phrase multiple strokes to one gesture, and separate the selection process from an operation that is applied to it [14]. Zeleznik & Miller [15] use terminal punctuation to define selection-action phrases. Several phrasing techniques (timeout, button, pigtail) where investigated by Hinckley et al. in [16, 17]. They also summarize that lassoing is the favored selection gesture "because it is well suited to selecting handwritten notes or diagrams, which typically contain many small ink strokes that would be tedious to tap on individually" [16, 17].

Interesting insight into the correlation of precision and quickness of selection gestures is given by Lank et al. [18]. Based on the motion dynamics of a gesture they made an "analysis of deliberateness versus sloppiness" and inferred "a linear relationship between tunnel width and speed, for paths of varying curvature" [18] which helps to automatically refine the selection.

3. TWO APPROACHES

The measure annotation task is a time-consuming process within the work on music editions. In practice, it includes a large number of interaction steps which have to be executed with high precision, putting a heavy load on the user. Hence, in our project we wanted to optimize the user interaction necessary for this task. We decided to follow two parallel paths: we included a measure annotation tool (Measure Editor) in the full featured music edition platform as a web application on one hand, and on the other hand implemented a native tablet tool (Vertaktoid) on an Android system for

[5] http://imslp.org, last access: Feb. 2017

the same purpose.

By doing so, we were enabled to compare the web-based solution to a solution dedicated to touch interaction on a popular device. Although the web-based application can be run on any device, it is primarily designed for the big screen on the desk, or even a multi-touch table, hence for a stationary work place. The tablet version on the other hand can be used in any mobile context and could, for instance, be used while working in a library or even during a musical performance.

The two tools worked in different contexts from the beginning:

- The Measure Editor is part of the music edition platform linked to an underlying data base (the back end), which is accessed via the front end representing the user interface. All semantic functions are triggered by the user from the front end. The input data for all operations come from the central data base and results are fed back into it. This includes, for instance, retrieval and storage of autographs, annotated autographs, concordances, and the complete set of possible FRBR [19] data. MEI imports and exports are also implemented through the data base, which implies that data exchange with Edirom is possible through this channel.

- Vertaktoid is a stand-alone prototype written explicitly for Android systems. As such, it can make use of native interaction techniques. The tool is not based on the central data base, but works with local storage on the device. Communication with other tools, including Edirom, is realized with MEI import and export functionality. MEI is also the internal format on which Vertaktoid operates directly.

While the Measure Editor is an integral part of the music edition, Vertaktoid could be taken as prototype to be linked into that environment as a practical tool for special purposes or usage situations. It could eventually work as a component of a distributed user interface approach, side by side with the web application Measure Editor and/or special purpose tools implemented for a multi-touch table. The developing of both tools is the main part of the contribution described in this paper.

3.1 Measure Editor

Measure Editor is a web-based application that is used for annotation of measures in music scores. The editor is supported by most popular browsers, such as Internet Explorer, Chrome, Opera, Firefox and Safari. The user interface has been designed allowing the user to work not only on the desktop, but also on any mobile device. A tablet with a stylus can be very suitable for precise work.

There are a lot of methods available for measure annotation task in the Measure Editor, which can be combined with each other. This provides especially quickly and easy annotation of measures of any shape. Since the measures mostly have a rectangular shape, the user always begins with bounding boxes. Measure Editor on desktop provides a spanning of rectangles by mouse as input device. The same task can be performed on the mobile devices by definition of the diagonal of rectangle. So, the user can create a new bounding box with two single clicks. Measures of regular rectangular shape can be easily transformed into polygons. It can be useful when the measure or another object with a complex geometric shape has to be marked. Figure 2 visualizes the transformation of a rectangle to polygon and back.

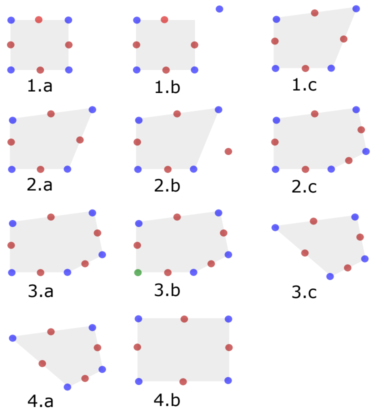


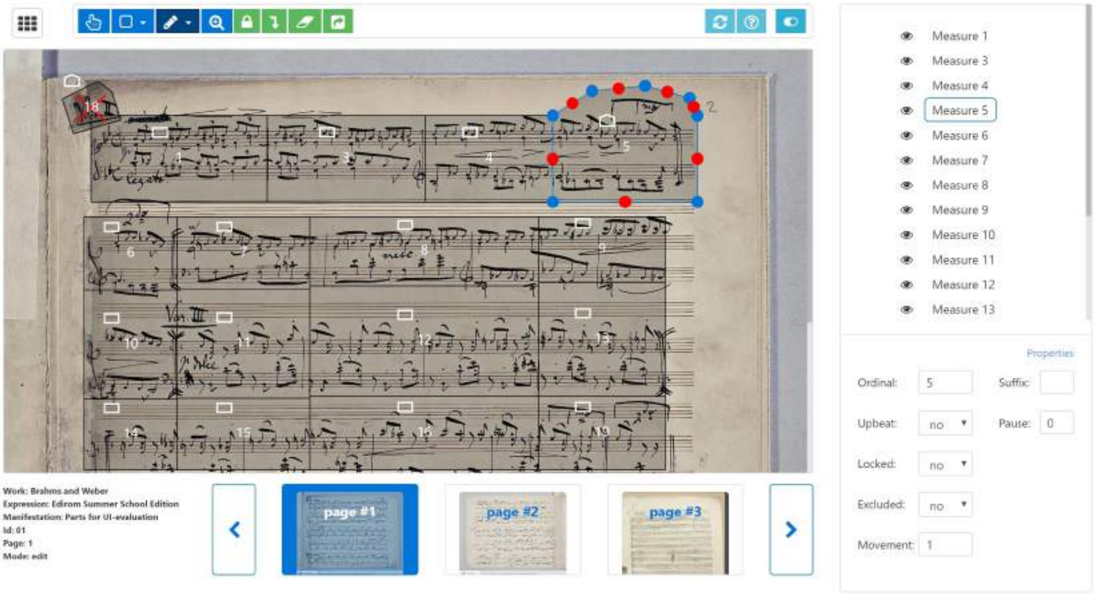
Figure 2. The transformation operations on the polygon.

Each rectangle consists of four vertices that are shown as blue dots. Red dots represent the midpoints of the edges. The user can change the shape and its size by moving its vertices. The first line in the Figure 2, 1.a - 1.c, demonstrates this effect. The drag & drop operation applied to midpoints will create new vertices (2.a - 2.c). The existing vertices can be removed by double-clicking with the condition that the number of vertices remains greater than or equal 3 (3.a - 3.c). The bottom row demonstrates the effect of converting a polygon to rectangle. Through this set of operations, the user can transform simple rectangles to complex geometric shapes.

Another equally important feature of Measure Editor is the pair of functions called *horizontal scissors* and *vertical scissors*. By means of these functions, the annotation of measures in music scores can be very quickly performed. The *scissors* function performs the division of rectangle into two smaller rectangles and so a new measure will be created. The division is either horizontally or vertically depending on the selected function. The new measure becomes automatically a number and inherits the attributes (e.g. movement alignment, suffix, upbeat) from the parent measure.

The GUI of Measure Editor is shown in Figure 3. The largest area is used for the representation of a score which is to be annotated. All functions available in a certain step are to be found in the toolbar. These are e.g. *select* , *move*, *draw*, *edit* a shape, *horizontal* and *vertical scissors*, *convert to rectangle*.

The right side of the GUI contains the list of all measures and the properties view. The user can select a specific measure graphically in the working area or directly as text in the list. The parameters of the selected measure can be set or adjusted in the properties view. For example, the



Work: Brahms and Weber
Expression: Edirom Summer School Edition
Manifestation: Parts for UI-evaluation
Id: 01
Page: 1
Mode: edit

Ordinal: 5
Suffix:
Upbeat: no
Pause: 0
Locked: no
Excluded: no
Movement: 1

Figure 3. User interface of Measure Editor.

user can change the ordinal number of a measure or the corresponding movement. The other parameters that can be adjusted are:

- *Suffix* - the additional textual information that can be added to the measure name after the ordinal number
- *Upbeat* - describes whether a measure is an upbeat of the parent movement.
- *Pause* - the number that represents the repeated rest in measure. It may be considered during calculating the sequence number of next measure and can have a meaning when defining concordances
- *Locked* - makes the measure uneditable.
- *Excluded* - marks the measure as excluded (not to be considered).

Measure Editor supports the music editors providing numerous features for comfortable and fast annotation of music scores. Inasmuch as measure annotation is a part of the whole process of music edition creation, Measure Editor will be also developed as a part of the complex system that was designed to provide different tools for music editors. Measure Editor is an open source application and is available [6] under LGPL 3.0 license.

3.2 Vertaktoid

Created as an android application for tablets with a pen, Vertaktoid wants to cover a need of mobile user-friendly instruments for comfortable and quickly annotation of measures in handwritten and printed music scores. As mentioned above, the measure annotation is an important and relatively time consuming part of whole creation process for digital music editions. Therefore, the main task of Vertaktoid is to minimize the man-hour costs and, on the other hand, to provide a comfortable interface for precise user-defined measure annotation process. Vertaktoid supports natively the MEI format and can be used in combination with Edirom Editor and Edirom Online programs. Vertaktoid is an open source application and was published [7] under LGPL 3.0 license.

3.2.1 Marking measures

To mark measures, the user selects the pen tool and draws an outline of the measure with the interactive stylus. The start of user's stroke is marked by a small circle. When the user closes the outline by reaching that circle again, the interaction is completed and the measure is defined by the rectangular bounding box of the outline. This interaction is very easy to understand and therefore well-suited for users that just start working with Vertaktoid. However, when marking several hundreds measures in a large music piece or even thousands of measures spread over several autographs, the efficiency of the interaction becomes much more important than its learnability. Therefore, Vertaktoid supports two further interaction methods:

- **"Click, don't draw":** If, while drawing the outline, the user lifts the stylus and puts it down at another position, the line from the last to the new position is automatically drawn. Now, the user can continue drawing the outline or click again at another position. When the user reaches the initial position (marked by the small circle) by clicking or drawing, the outline is completed.
- **"Cut, don't repeat":** Oftentimes the user marks several measures that are aligned as they are part of the same grand staff (see, e.g., measures 1-4 in Figure 4).

[6] `https://bitbucket.org/zenmem/zenmem-frontend`, last access: May 2017

[7] `https://github.com/cemfi/vertaktoid`, last access: May 2017

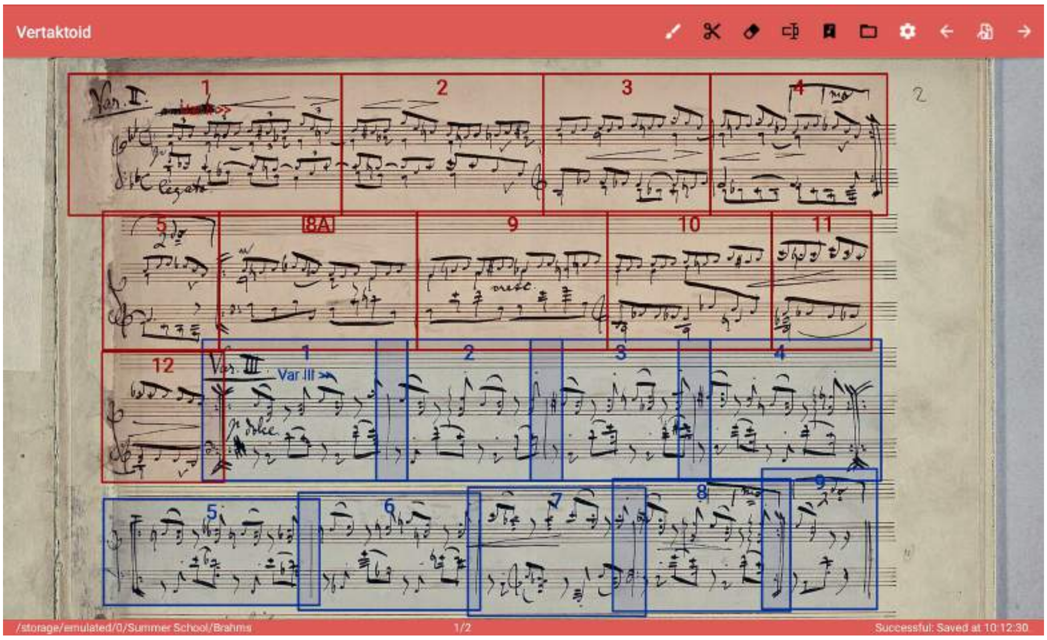

Vertaktoid
1
2
3
4
5
8A
9
10
11
12
Var III
1
2
3
4
5
6
7
8
9
/storage/emulated/0/Summer School/Brahms
1/2
Successful: Saved at 10:12:30

Figure 4. User interface of Vertaktoid.

> Instead of repeating and drawing the outline for each measure separately, the user can outline the entire grand staff and then use the scissors tool to cut the outline in pieces by clicking at the measure bars.

By clicking and cutting, the user can perform the task considerably faster. Consider, e.g., the red Measures 1-4 in Figure 4. With clicking and cutting, a user would typically use ten clicks to mark those four measures: Five clicks would be needed to mark the four corners of the grand staff and close the shape by clicking on the first corner for a second time. Three clicks would be needed for cutting. Further two clicks would be needed for first selecting the pen and then switching to the scissors tool. Sometimes the vertical alignment of the measures is not wanted.

Some music editors prefer to include a small horizontal overlap from the previous and the next measure. This way, the measure does not look isolated when, e.g., a musician views an annotation by the music editor which refers to that measure. Sometimes an overlap is not purely aesthetic but necessary in order to understand the content of the measure, e.g., an overlap may be necessary if a sustained note is written with a tie across the bar line. In order to enable the use of the scissors tool in such situations, the user can configure the amount of horizontal overlap. While the blue Measures 1–4 in Figure 4 were made with an overlap, the red measures were made without. Also if the scissors tool is not able to create the desired result, the user can always define the measure outlines manually (see Figure 4, blue Measures 5–9).

3.2.2 Automatic measure numbering

Another feature of Vertaktoid is automatic measure numbering and coloring. The latter indicates the movement the measure belongs to. The measure numbers are calculated through comparison of the measure positions on the facsimile page and their vertical overlapping with each other.

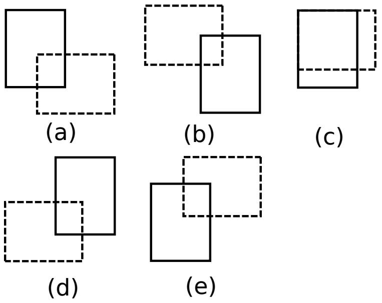


Figure 5. Five cases by measure positions comparison.

When a new measure is added, it is compared to all other measures on the page. In Figure 5, the new measure is depicted as the dashed rectangle while the solid rectangle represents an already existing measure. Figure 5 demonstrates the five possible cases that can occur.

(a) The new measure is located vertically and horizontally after the existing measure.

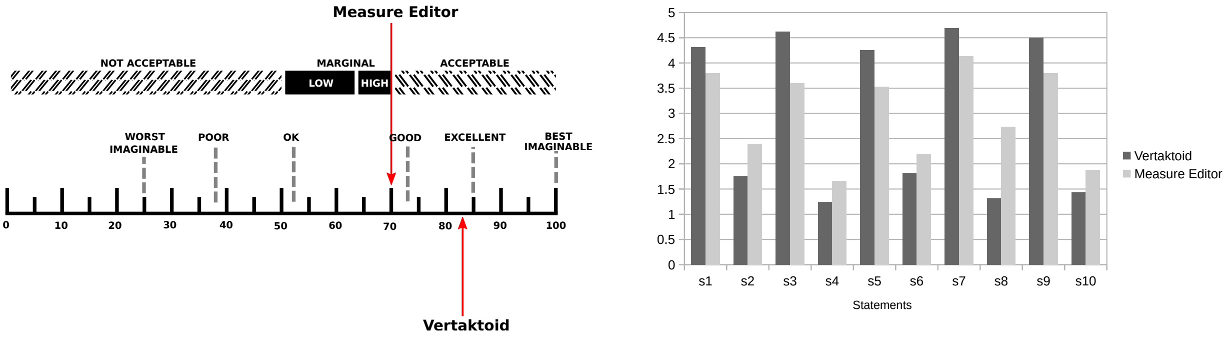
(b) The new measure is located vertically and horizontally before the existing measure.

(c) The new measure and the existing measure have the same start positions.

(d) The new measure lies vertically after and horizontally before the existing measure.

(e) The new measure lies vertically before and horizontally after the existing measure.

The first three cases are trivial. In case (a) the new measure will be ordered after the existing, in case (b) - before the existing and in case (c) - the both measures have equal start positions and can be randomly ordered.



(a) The meaning of the SUS Score depicted on adjective ratings and acceptability ranges.

(b) Average spreading of participants votes over SUS statements.

Figure 6. The results of SUS survey.

For the last two cases, a calculation of the vertical overlap is needed. Let M be the rectangle that represents the existing measure and M' - the rectangle that represents the new measure. The rectangles can be defined by two points: ul - upper lefter vertex and lr - lower righter vertex. Each vertex consists of x and y coordinates. Then the vertical overlapping $Over_y$ is calculated as follows:

$$Over_y = \frac{\min{(lr_y^M, lr_y^{M'})} - \max{(ul_y^M, ul_y^{M'})}}{\min{(lr_y^M - ul_y^{M'}, lr_y^M - ul_y^{M'})}} \qquad (1)$$

$Over_y$ is the vertical intersection of both rectangles relative to the height of the smaller shape. Rectangles, which doesn't have a horizontal intersection, can be also processed. Now that the vertical overlapping factor is calculated, the final order of two examined measures can be decided. The algorithm will take the horizontal order of measures if $Over_y \geq 0.5$ and the reverse horizontal order otherwise. This approach is applied to each measure from the corresponding movement until the measures are sorted by their positions.

Sometimes, such a strict numbering scheme is not appropriate for the musical content, e.g., for upbeat measures, multiple measure rests or measures that are split between two successive pages or staves. In those cases the user can manually assign an arbitrary name for the measure. Such names are marked with a frame (see measure "8A" in Figure 4). The alignment of measures to the movements can be also changed. Both user-performed operations causes further automatically renumbering of following measures inside of the corresponding movements.

4. EVALUATION & DISCUSSION

To evaluate the usability of both measure annotation tools, a quantitative study was carried out. The study was based on the System Usability Scala (SUS) [20]—a simple and quick survey that can be used to measure the usability of a product or service based on the subjective assessments of participants. The SUS survey contains ten statements, which must be voted with a value in the range from 1 (strongly disagree) to 5 (strongly agree). The statements alternate between positive and negative expressions about the examined software. A simple formula calculates the SUS score from the individual votes. Figure 6(a) shows the meaning of a SUS score translated into adjective ratings [21].

Our study was conducted as part of the Edirom Summer School event [8], which was created to teach musicologists the creation of digital music editions using the Edirom Editor tool and to inform experienced music editors about the new techniques in this field. We recruited sixteen music editors for our study. The familiarity of participants with the music annotation software was heterogeneous: some of them have learned the Edirom Editor during the Summer School, the other part have already a good expirience with it. They used prototype versions of Measure Editor and Vertaktoid and annotated autographs with them. Figure 6 shows the average votes for both applications. The resulting SUS scores were 70 for Measure Editor and 87 for Vertaktoid. This means that both tools were found as acceptable and Vertaktoid even as excellent (see Figure 6(a)).

To decide about future development, we interviewed the study participants. The combination of the survey and these consultations was taken into consideration during the developing of next versions for both applications. Their feedback contains a lot of clever suggestions. So, the cut function can be performed in both dimensions, what can make the annotation process more effective. Some users found the often repeating selection of suitable control elements as uncomfortable. The next control element in some cases can be automatically chosen following the common workflow. The whole annotation process can be also performed in the predefined sequence of steps. Thus, the application can offer the user to annotate the measures with draft bounding boxes that can be refined during the next step.

Furthermore, Vertaktoid is currently used at Detmolder Hoftheater project, which provides continuous helpful feedback for further evaluation and troubleshooting.

A number of features could be added in the future. Optical measure recognition and subsequent automatic annotation would be a helpful and time-saving function at least for annotation of printed music sheets. It would also be productive if several users could simultaneously work on one music score. This would mean that users could access data from a shared database to continue the work of another

[8] `http://ess.uni-paderborn.de`, last access: May 2017

users, make corrections or discuss further work flow. It would also be useful if the users could make additional textual (and not only textual) annotations for individual measures, music notes or even any other positions on the facsimile. These annotations could be considered in the later steps during the creation of a music edition.

Both tools can also contribute to each other and, using the experience of the sibling tool, inherit its best techniques. Thus, the possibility to annotate an area via polygon can be very helpful in Vertaktoid as well as the editing functions. On the other hand, Measure Editor can be improved by additional interaction techniques, movement colorizing or automatic numbering.

5. REFERENCES

[1] M. Reger, *Reger-Werkausgabe, Bd. I/1: Choralphantasien für Orgel*, A. Becker, S. König, C. Grafschmidt, and S. Steiner-Grage, Eds. Leinfelden-Echterdingen, Germany: Carus-Verlag, 2008.

[2] P. Roland, "The Music Encoding Initiative (MEI)," in *Proceedings of the International Conference on Musical Applications Using XML*, 2002, pp. 55–59.

[3] A. Hankinson, P. Roland, and I. Fujinaga, "The music encoding initiative as a document-encoding framework," in *Proceedings of the 12th International Society for Music Information Retrieval Conference*, Miami (Florida), USA, October 24-28 2011, pp. 293–298.

[4] Virtueller Forschungsverbund Edirom (ViFE), "Edirom Editor 1.1.26," https://github.com/Edirom/Edirom-Editor, June 2016, accessed: 2017-20-02.

[5] Virtueller Forschungsverbund Edirom (ViFE), "Edirom Online," https://github.com/Edirom/Edirom-Online, accessed: 2017-20-02.

[6] I. Capelle and K. Richts, "Cataloguing of musical holdings with MEI and TEI. The Detmold Court Theatre Project," in *Music Encoding Conference 2015*. Florence, Italy: Music Encoding Initiative, May 2015.

[7] S. Cox, M. Hartwig, and R. Sänger, "Beethovens Werkstatt: Genetische Textkritik und Digitale Musikedition," *Forum Musikbibliothek*, vol. 36, no. 2, pp. 13–20, July 2015.

[8] A. Rebelo, I. Fujinaga, F. Paszkiewicz, A. R. S. Marcal, C. Guedes, and J. S. Cardoso, "Optical music recognition: state-of-the-art and open issues," *International Journal of Multimedia Information Retrieval*, vol. 1, no. 3, pp. 173–190, 2012.

[9] C. Dalitz and T. Crawford, "From Facsimile to Content Based Retrieval: the Electronic Corpus of Lute Music," *Phoibos – Zeitschrift fur Zupfmusik*, pp. 167–185, February 2013.

[10] G. Vigliensoni, G. Burlet, and I. Fujinaga, "Optical Measure Recognition in Common Music Notation," in *Proceedings of the 14th International Society for Music Information Retrieval Conference, ISMIR 2013, Curitiba, Brazil, November 4-8, 2013*, 2013, pp. 125–130.

[11] D. Bainbridge and T. Bell, "The challenge of optical music recognition," *Computers and the Humanities*, vol. 35, no. 2, pp. 95 – 121, 2001.

[12] V. Padilla, A. McLean, A. Marsden, and K. Ng, "Improving optical music recognition by combining outputs from multiple sources," in *Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR 2015, Málaga, Spain, October 26-30, 2015*, 2015, pp. 517–523.

[13] E. Artinger, T. Coskun, M. Schanzenbach, F. Echtler, S. Nestler, and G. Klinker, "Exploring Multi-touch Gestures for Map Interaction in Mass Casualty Incidents," in *INFORMATIK 2011 – Informatik schafft Communities (Proc. of the 41st GI-Jahrestagung)*, H.-U. Heiß, P. Pepper, H. Schlingloff, and J. Schneider, Eds., Gesellschaft für Informatik (GI). Berlin: Bonner Köllen Verlag, Oct. 2011.

[14] K. Hinckley, F. Guimbretiere, M. Agrawala, G. Apitz, and N. Chen, "Phrasing Techniques for Multi-Stroke Selection Gestures," in *Proc. of Graphics Interface 2006*. Canadian Human-Computer Communications Society, June 2006, pp. 147–154.

[15] R. Zeleznik and T. Miller, "Fluid Inking: Augmenting the Medium of Free-Form Inking with Gestures," in *Proc. of Graphics Interface 2006*. Canadian Human-Computer Communications Society, June 2006, pp. 155–162.

[16] K. Hinckley, P. Baudisch, G. Ramos, and F. Guimbretiere, "Design and Analysis of Delimiters for Selection-Action Pen Gesture Phrases in Scriboli," in *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems (CHI 2005)*. Portland, Oregon, USA: ACM SIGCHI, Apr. 2005, pp. 451–460.

[17] Hinckley, K. and Baudisch, P. and Ramos, G. and Guimbretiere, F., "Delimiters for Selection-Action Pen Gesture Phrases," United States Patent No. US 7,454,717 B2, USA, Nov. 2008, filed Oct. 2004.

[18] E. Lank, E. Saund, and L. May, "Sloppy Selection: Providing an Accurate Interpretation of Imprecise Selection Gestures," *Computers & Graphics*, vol. 29, no. 4, pp. 490–500, Aug. 2005.

[19] B. Tillett, "What is FRBR? a conceptual model for the bibliographic universe," Library of Congress, Tech. Rep., February 2004.

[20] J. Brooke, "SUS: a "quick and dirty" usability scale," in *Usability Evaluation In Industry*, P. W. Jordan, I. L. McClelland, B. Thomas, and B. A. Weerdmeester, Eds. London: Taylor and Francis, 1996, pp. 189–194.

[21] A. Bangor, P. Kortum, and J. Miller, "Determining What Individual SUS Scores Mean: Adding an Adjective Rating Scale," *J. Usability Studies*, vol. 4, no. 3, pp. 114–123, May 2009.

Renotation of Optical Music Recognition Data

Liang Chen, Christopher Raphael
School of Informatics and Computing, Indiana University Bloomington
chen348@indiana.edu, craphael@indiana.edu

ABSTRACT

We present the problem of music renotation, in which the results of optical music recognition are rendered in image format, while changing various parameters of the notation, such as the size of the display rectangle or transposition. We cast the problem as one of quadratic programming. We construct parameterizations of each composite symbol expressing the degrees of freedom in its rendering, and relate all the symbols through a connected graph. Some of the edges in this graph become terms in the quadratic cost function expressing a desire for spacing similar to that in the original document. Some of the edges express hard linear constraints between symbols expressing relations, such as alignments, that must be preserved in the renotated version. The remaining edges represent linear inequality constraints, used to resolve overlapping symbols. The optimization is solved through generic techniques. We demonstrate renotation on several examples of piano music.

1. INTRODUCTION

A symbolic music encoding, such as MEI [1], MusicXML [2], and many others, represents music in a hierarchical format, naturally suited for algorithmic manipulation. One of the most important applications of symbolic music encodings will likely be what we call *music renotation* [3], which renders an image of the music notation, perhaps modified in various ways. For instance, music renotation may produce a collection of parts from a score, show the music in transposed form, display notation expressing performance timing, or simply display the music in an arbitrarily-sized rectangle. Music renotation may someday be the basis for digital music stands, which will offer a range of possibilities for musicians that paper scores do not, such as instant and universal access to music, page turning, search, and performance feedback. While these ideas could apply to any type of music notation, our focus here is on common Western notation.

As long as there has been interest in symbolic music representations, there has been a parallel interest in rendering these encodings as readable music, e.g. [4], [5]. There is a well-developed craft surrounding this engraving problem, predating algorithmic efforts by centuries. The essential task is to produce music documents that are easy to read and make efficient use of space. Often the engraver must choose between many possible equivalent representations of the music, while considering the placing of symbols, avoiding unnecessary overlap, preserving important alignments between symbols, and leading to a pleasing document overall.

Our essential idea is to leverage the choices made in a previous engraving to guide our renotated rendition. This includes the many *hard* or categorical decisions made in the original document, such as what notes should be beamed, what stem directions to use, as well as the many other choices that lead to equivalent notation in terms of pitch and rhythm. However, we also seek to leverage the specific *soft* choices of layout and spacing that give the music the desired density of information while leading to high readability. In essence, our renotated music seeks to copy as much of the original layout as possible, while simultaneously satisfying the constraints imposed by our renotated format, such as the dimensions of the display rectangle.

Our formulation of the renotation problems ties in naturally to our work in optical music recognition (OMR). Our *Ceres* system [6–8] is a longstanding research interest that seeks to recognize the contents of a music document. From the standpoint of recognition, it is almost impossible to identify a musical symbol in an image without also knowing the precise location and parametrization of the symbol, thus accurately registering the recognized symbol with the image data. Therefore, a welcome consequence of OMR is a precise understanding of the music layout, including both the *hard* and *soft* choices, leaving our current effort poised for high-quality renotation. This is the goal we address here.

Our essential approach is to cast the renotation problem as one of *optimization*, expressed in terms of a graph that connects interdependent symbols. This view is much like the spring embedding approach to graph layout, so popular in recent years [9–12]. Spring embeddings seek to depict the nodes and edges of a graph in a plane, with minimal conflict between symbols, while clustering nearby nodes. Unlike the graph layout problem, there is no single correct graph for music renotation. Rather, the construction of an appropriate graph lies at the heart of the modelling challenge. Our work bears some resemblance to the work of Renz [13], who takes a spring embedding view of one-dimensional music layout.

2. MODELING THE NOTATION

At present, we do not model the entire collection of possible musical symbols, limiting ourselves to the most common ones, though providing generic mechanisms to extend to larger collections. This restriction of attention parallels the state of our OMR system, which recognizes only a subset of possible notation, at present. It seems that any music notation effort must eventually come into contact with the "heavy tail" problem — there are a great many unusual symbols, special cases, and exceptions to familiar rules which, collectively, pose a formidable modeling challenge [14]. To avoid getting buried in the many details of real-life music notation we focus here on the *core* of musical symbols, by which we mean those involved in conveying pitch and rhythm (beams, flags, stems, note heads, ledger lines, accidentals, clefs, augmentation dots, time signatures, key signatures, rests, etc.), with a few additions to this core.

We divide the world of music symbols into *composite* and *isolated* symbols. The isolated symbols, (clefs, accidentals, etc.), are stand-alone symbols, represented by single characters in a music font. The composite symbols, including beams and chords, are formed by constrained configurations of *primitives*, such as beams, stems, heads, ledger lines, flags, etc. For purposes of renotation we view the *satellites*, (accidentals, articulations, augmentation dots, etc.) as isolated symbols, rather than as parts of a composite symbol.

When drawing a composite symbol one must obey basic constraints between the constituent primitives. For instance, the vertical positions of the note heads are fixed in relation to the staff, while their horizontal positions are fixed in relation to the note stem. These relations hold with both regular and "wrong-side" note heads. After accounting for these constraints, the only degrees of freedom in drawing a chord (not including its satellites) are the horizontal and vertical locations of the stem endpoint. The locations of all other primitives follow deterministically once the stem endpoint is fixed. Beamed groups have more degrees of freedom in their rendering. After fixing the corners of the beamed group one must also decide upon the horizontal positions of the stems, leading to $n + 2$ degrees of freedom for an n-note beamed group. With these parametrizations in mind, we can think of all symbols, composite and isolated, as parametrized objects, where an isolated symbol is parametrized simply by its two-dimensional location. In some cases, one coordinate of an isolated symbol may be fixed, as with the vertical coordinate of an accidental or clef. In such cases the symbol has a one-dimensional parameter.

2.1 Overview of Approach

We cast the renotation problem as constrained optimization, in particular, quadratic programming (QP) [15], in which one minimizes a quadratic function subject to linear equality and inequality constraints. To this end we let $v_1, \ldots, v_N$ be the musical symbols, isolated *and* composite, and let $x = (x_1, \ldots, x_N)$ be the parameters for these symbols. Thus the notation is completely determined by x. We define a quadratic objective function, $Q(x)$, measuring the desirability of the parameter x. Q penalizes the degree to which the renonated symbols' relative positions differ from those in the original notation — we simply try to copy the original notation as much as possible. To capture hard constraints, such as the essential symbol alignments between rhythmically coincident notes, we include a collection of linear equality constraints $\{a_m^t x = b_m\}_{m=1}^M$, which we will summarize as $Ax = b$. To capture the non-overlapping constraints, we include inequality constraints of the form $\{a_{m'}^t x \le b_{m'}\}_{m'=1}^{M'}$ summarized as $A'x \le b'$. Thus we define our desired solution as

$$\hat{x} = \arg \min_{x: Ax=b, A'x \le b'} Q(x) \qquad (1)$$

QP gives us a formulaic way of solving an optimization problem, once posed in this manner.

To be more specific, we define our objective function and constraints in terms of a graph, $G = (E, V)$. In our graph, the parametrized symbols become the vertex set, $V = \{v_1, \ldots, v_N\}$. We define an edge set composed of three types of edges between these vertices, *soft*, *hard* and *conflict*, denoted by $E = E_s \cup E_h \cup E_c$. Our graph will be *connected* by the $E_s \cup E_h$ edges, thus ensuring that there are no notational "islands" that function independently of the rest. For every edge, $e = (v_i, v_j) \in E_s$ we define a quadratic term, $q_e(x_i, x_j)$, so that our objective function, Q, decomposes as

$$Q(x) = \sum_{e=(v_i,v_j)\in E_s} q_e(x_i, x_j)$$

Similarly, for each edge $e = (v_i, v_j) \in E_h$ we define a constraint, $a_e^t x = b_e$, where the vector, a_e, is non-zero only for the components corresponding to x_i and x_j. Analogously we have a constraint, $a_e^t x \le b_e$ for each edge $e \in E_c$. Thus A and b from Eqn. 1 become

$$A = \begin{pmatrix} a_{e_1}^t \\ a_{e_2}^t \\ \vdots \\ a_{e_M}^t \end{pmatrix} \qquad b = \begin{pmatrix} b_{e_1} \\ b_{e_2} \\ \vdots \\ b_{e_M} \end{pmatrix}$$

where $e_1, \ldots, e_M$ is an arbitrary ordering of the edges of E_h, with a similar definition for A' and b'.

2.2 A More Detailed Look: Beamed Groups

We examine here the case of a beamed group in more detail, so as to flesh-out our ideas. As already described, the beamed group itself (without its satellites) is parametrized by an $(n + 2)$-dimensional parameter, x_b. Each note on each chord of the beamed group may have an associated accidental, while the heights of these accidentals are fixed at the appropriate staff position. Thus we can write $\{x_i^{\text{acc}}\}$ for the the horizontal positions of these accidentals, indexed by i. Absent other considerations, there is an ideal horizontal distance between a note head and its accidental. However, in some cases, particularly with a densely-voiced chord, the accidentals may be placed far to the left of this ideal to avoid overlap. For each accidental we introduce a quadratic term, $q_i^{\text{acc}}(x_b, x_i^{\text{acc}})$, that penalizes the degree that the accidental displacement differs from that in the original image. We also must deal with potential overlap between the accidentals, and will return to this issue in what follows.

Augmentation dots are more variable in their vertical placement than are accidentals. In part, this results from the need to avoid conflicts with the staff lines, however, densely-voiced chords often require that augmentation dots be notated far away from their nominal vertical positions to avoid conflicts. We introduce quadratic terms $q_i^{k,\text{aug}}(x_b, x_i^{k,\text{aug}})$ for $k = 1, 2$ where $x_i^{k,\text{aug}}$ is the kth coordinate of the ith augmentation dot position. Again, this term penalizes the difference between the actual displacement from the note head and that observed in the original image. If each member of a chord has a single augmentation dot, those augmentation dots are constrained to share a single horizontal position by adding appropriate linear equality constraints, proceeding analogously with additional augmentation dots.

Note heads may have one or several symbols above (stem-up) or below (stem-down) the note head, such as articulations, ornaments, fingerings, etc. We will refer to these symbols collectively as "markups." The case of a single markup is easy to handle, as such a symbol is usually centered over the note head, thus constraining its horizontal position. This situation is handled in analogy with the accidental, in which we seek to replicate the vertical separation found in the original image while constraining the relative horizontal position to the note head. Occasionally a note head has several associated markup symbols, requiring that we *infer* the desired constraints. Often markups are stacked, which calls for a shared horizontal position with all of the symbols (equality constraints). Though sometimes one sees more elaborate configurations calling for different kinds of alignment. For instance, one may have a trill with two numbers above the trill indicating the fingers to be used. Typically the pair of finger numbers would be centered over the trill, calling for a different collection of constraints. Suffice it to say that when multiple markups are present we must infer the desired equality constraints before we impose these.

2.3 Conflict Resolution

When we view a beamed group in isolation we would expect no conflicts between the various satellites surrounding the note heads: their preferred relative positions are those from the original image, which presumably do not overlap. However, we now view a beamed group as a *flexible* object having desired relationships with the *other* symbols composing the notation. These other symbols exert influence on the beamed group. As we allow the beam group to be deformed away from its original presentation, we should expect conflicts to develop between the symbols as they inadvertently overlap one another.

We make two observations regarding symbol conflicts that will guide our process for resolving them, not just with beamed groups, but all conflicts that can arise. Consider, first, the case of two rectangular bounding-box regions for symbols v_i and v_j, $(a_l^i, a_h^i) \times (b_l^i, b_h^i)$ and $(a_l^j, a_h^j) \times (b_l^j, b_h^j)$, both expressed as the cross product of horizontal and vertical intervals, as in Figure 1. There are four relevant linear inequality constraints, the satisfaction of any one of these ensuring the regions do not overlap:

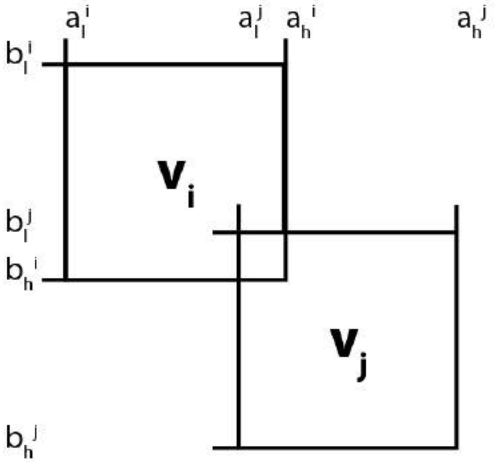


Figure 1. A conflict between two symbols, v_i and v_j, can be resolved by moving the relative positions of their bounding boxes in one of four ways, illustrated by the four dotted rectangles.

$$\begin{aligned} a_l^i &\geq a_h^j \\ a_h^i &\leq a_l^j \\ b_l^i &\geq b_h^j \\ b_h^i &\leq b_l^j \end{aligned}$$

These correspond to moving v_j left, right, up, or down, relative to v_i. There is no general way to express a disjunction of linear inequality constraints as a conjunction of linear inequality constraints, thus we cannot, with QP, perfectly model the concept that a pair of symbols cannot overlap. We resolve overlap between a symbol pair by selecting *one* of the above four constraints to include in the QP formulation.

The second observation is that it isn't necessary to include all inequality constraints in the initial statement of our optimization problem. That is, suppose we minimize the objective function, $Q(x)$, subject only to the equality constraints. If the resulting configuration happens to satisfy the inequality constraints, then we have found the fully constrained (equality and inequality) optimum as well.

We extend this reasoning for our problem. While there are, in principle, non-overlapping (inequality) constraints between any pair of symbols in our notation, we do not impose these at first, instead solving a simpler QP problem that allows the symbols to overlap. If the solution of this problem contains overlap, we impose additional constraints to alleviate this problem. In particular, for each overlapping symbol pair, we choose one of the inequality constraint from the four constraints enumerated above that is near to being satisfied. We further restrict our choice of constraint to be *consistent* with the original notation. For instance, if two symbols, v_i and v_j overlap but v_j lies to the right of v_i in the original notation, we could choose the constraint $a_h^i \leq a_l^j$. This constraint would be expressed in terms of the parameters x_i, x_j, and included into the set of inequality constraints. This approach guarantees that our QP problem has a non-trivial feasible solution, since the layout of the original image is a feasible point. If v_i and v_j overlap in the *original* image, we cannot resolve this situation, though such situations are uncommon and usually occur in layout situations without obvious solutions.

We iterate the process of solving a QP problem, and imposing new constraints to resolve any newly-discovered overlapping symbols. As all of the newly-added inequality constraints are consistent with the original notation, the QP problem has a feasible region, thus remains well-posed. Our final result is a solution to the QP problem with equality constraints, plus the additional constraints imposed through the iterative process. In this way we *approximate* the notion of non-overlapping constraints.

Our approach constrains us to solve layout problems in a way that is similar to what has been done in the original notation, rather than seeking novel ways of resolving conflicts. In essence, this is consistent with the heart of our proposed approach that leverages the existing notation by deferring to the original notation. The problem of constructing good notation from a symbolic music representation containing no positional information is considerably more challenging, and, perhaps, unnecessary.

2.4 Constructing the Symbol Graph

We have reduced the modeling problem to constructing a graph on the symbol vertices with edges labeled as either *soft*, *hard* or *conflict*, corresponding to quadratic terms, equality constraints, and inequality constraints. The graph for each system first begins by ordering the *basic* symbols according to their left-to-right presentation in the original image. These *basic* symbols include all notes, rests, clefs, key signatures symbols, and bar lines. These are connected in left-to-right manner with *soft* edges derived from the original image. While a beamed group is composed of a collection of notes, the edges are between the notes themselves, reflecting our desire to space approximately as done in the original. These edges are only concerned with horizontal distance. The other type of soft edges are used to express the desired vertical distance such as the stem length and the distance between markups and their associated note heads.

In addition, we introduce *hard* edges between various pairs of symbols. There are several situations that give rise to these equality constraints. For instance, "opposing chords" are pairs of "stacked" chords with a stem-up chord on top and a stem-down chord on bottom. These are recognized as single symbols by our OMR system, thus our representation implicitly understands the need for the alignment constraint. We add hard edges between such chord pairs to force their horizontal alignment. Other symbol alignments are detected by thresholding the actual positions of symbols, while these are also modeled with hard edges that force their horizontal alignment. Our eventual goal is to have the process entirely determined through interpretation of the symbols' meaning, for instance rhythm recognition to determine simultaneous notes, though we approximate this goal, at present, through thresholding.

While our analysis is presently limited to these basic symbols, we can extend to a greater variety of symbols, including slurs, text, dynamics, etc., by "anchoring" to the basic symbols. In the most common case, such a symbol is owned or related to a basic symbol. For instance, a dynamic symbol or text may be associated with a certain note, requring its horizontal placement to align with the note. A great variety of symbols can be included into the present context by aligning them to the basic symbols with hard edges that force the appropriate horizontal alignment while using a quadratic term to encourage the vertical spacing in the original image.

Figure 2 shows a graph that is constructed from the notation, expressing the optimization problem we solve. Different colored edges in this figure correspond to different types of quadratic penalty, alignment, and overlap constraint. As outlined above, we proceed by solving the optimization problem defined by the hard and soft edges first. In doing so, conflicts will arise between symbols that overlap in the "optimal" result. We then iterate the process of imposing conflict edges (inequality constraints) between the overlapped symbols until none remain. These conflict edges, as well as the other types of edge are indicated in the figure.

3. RENOTATION EXPERIMENTS

Here we present preliminary experiments demonstrating our proposed ideas in several renotation scenarios. All the image data, graphical models and generated notations in our experiments can be accessed from the website: `http://music.informatics.indiana.edu/papers/smc17/`. As mentioned before, we treat only a subset of the possible realm of music notation, in an effort to develop an overall approach without being hampered by the many complexities of real-life notation. In particular, we restrict our attention to the rhythm and pitch bearing symbols. We believe the basic framework we have established extends naturally to more complex notation.

Our first experiment displays music notation in *panorama mode*, in which the image is composed of a single long rectangle containing only a single system. This kind of display avoids line breaks, so it is easier to produce as it doesn't need to accommodate the notion of a *page*. Some people may prefer the simplicity of panorama notation in some situations, though the strengths and weaknesses of this notation mode are beyond the scope addressed here.

Panorama mode requires that we delete clef-key-signatures that usually begin each line of music, as the single display line makes these redundant. Otherwise, the display layout follows the approach described above. As the bounding box for the notation doesn't lend itself to the pages of a conference proceedings paper, we include links to two examples. The first example shows a page of the first movement of Mozart's Piano Sonata, K. 333, with the notation recognized and renotated in panorama mode from the

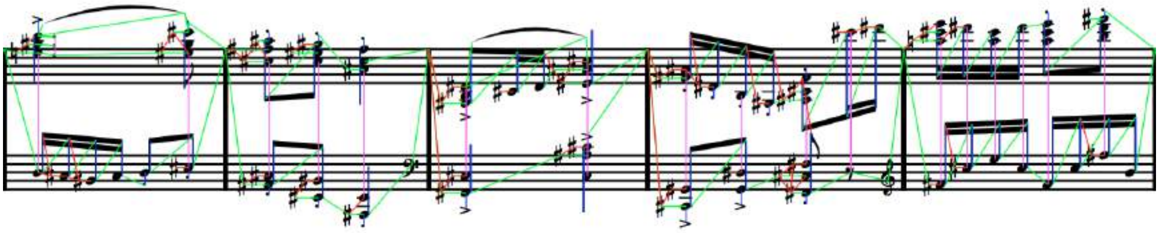


Figure 2. Graph showing the types of edges between the vertices. *Green:* Soft horizontal edges (quadratic term); *Blue:* Soft vertical edges (quadratic term); *Magenta:* Hard Edges (equality constraint); *Red:* Conflict edges (inequality constraint).

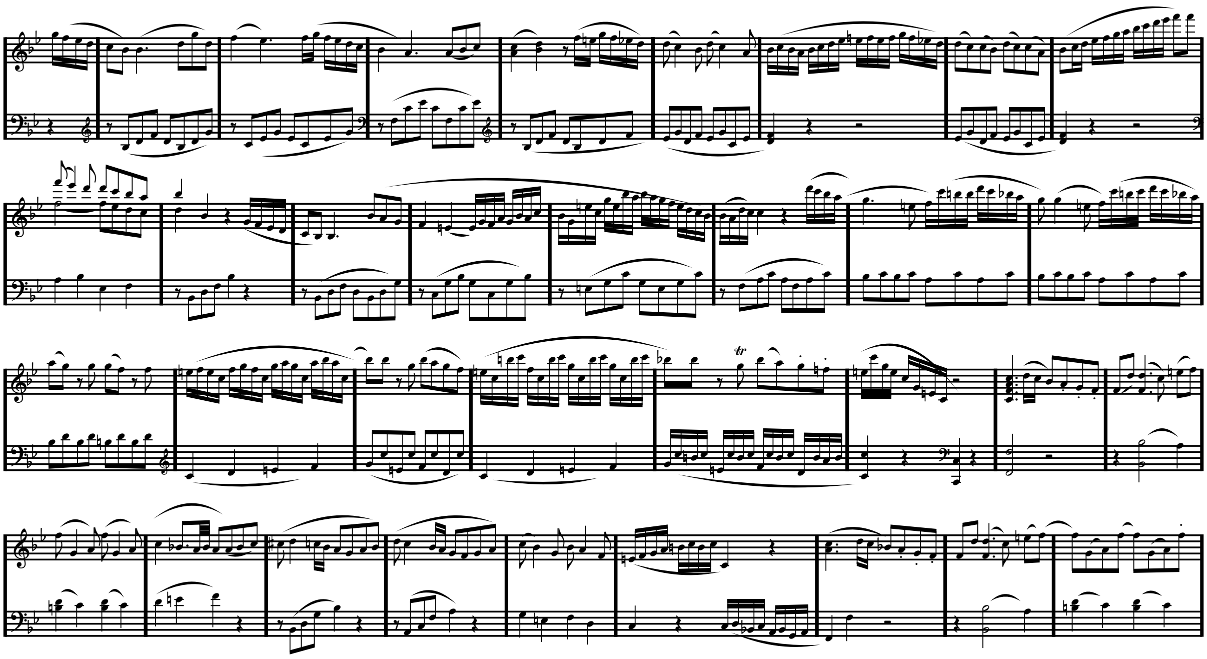


Figure 3. A section from the first movement of Mozart's Piano Sonata, K. 333 (Breitkopf and Härtel 1878 edition)

1878 Breitkopf and Härtel edition [1]. As with this example, many of the most useful public domain editions are quite old, though these editions are still excellent sources for renotation, with a significant degree of expertise and care supporting the engravings. The second shows a page of the Rigaudon from Ravel's *Le Tombeau de Couperin*, using the Durand 1918 edition [2].

Obviously the notation for these examples appears spare without the additional symbols that describe style, dynamics, articulations, etc., though one can still appreciate the value of the proposed ideas based on these examples. While some spacing issues are not resolved in the most readable manner, the overall notation manages to preserve the appropriate rhythmic alignment and produce somewhat natural-looking notation in a *fully automatic* way.

The second pair of experiments show the same two pieces, now notated in page mode, in Figures 3 and 4. The page sizes were chosen to be considerably wider than the original, thus forcing our algorithm to adjust the spacing to achieve the customary alignment of the rightmost bar line of each system. In addition to removing the original clef-key-signatures at the start of each original staff line, we added the appropriate key-clef-signatures at the beginnings of our renotated staff lines.

4. CONCLUSIONS AND FUTURE WORK

Our renotation examples demonstrate work in progress, thus there are still some issues that need to be resolved in the definition of our objective function before the notation has a professional, easy-to-read look. For instance, smaller symbols, such as accidentals and articulations have different requirements for spacing, and are not ideally han-

[1] http://music.informatics.indiana.edu/papers/smc17/Mozart_panorama.pdf

[2] http://music.informatics.indiana.edu/papers/smc17/Ravel_panorama.pdf

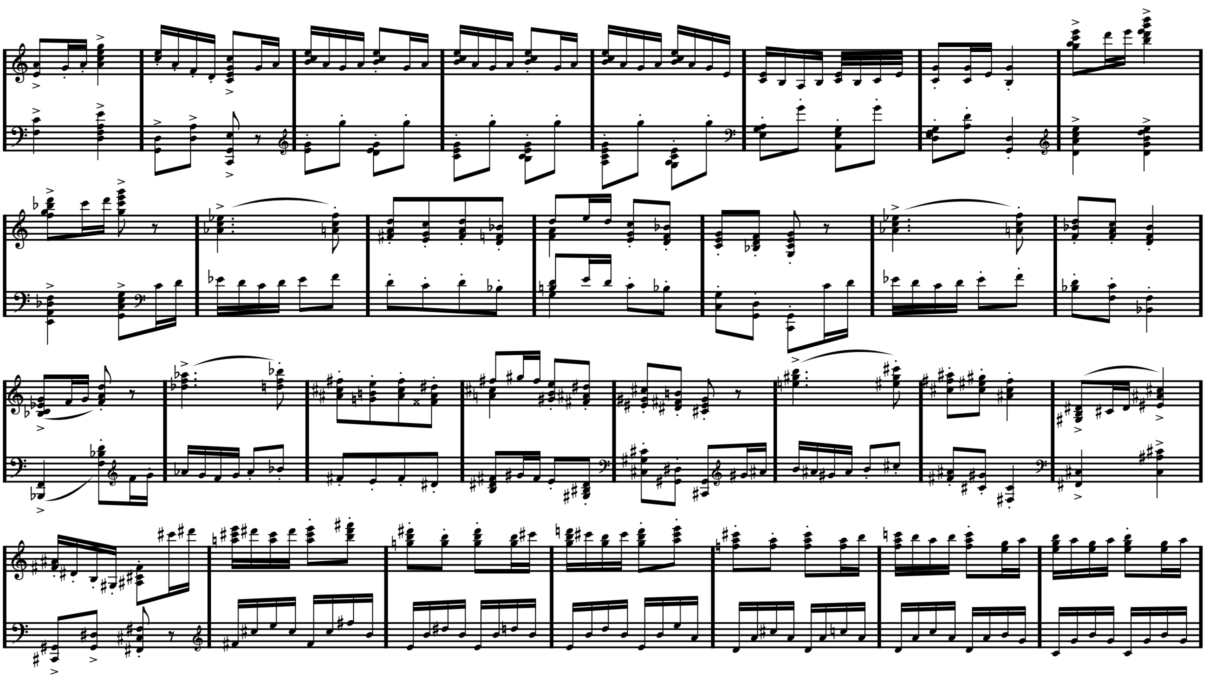


Figure 4. A section from the Rigaudon from Ravel's *Le Tombeau de Couperin*, (Durand and Cie. 1918 edition)

dled by the more generic approach to spacing we take at present.

There are some aspects of this approach that are novel and worth emphasizing. First of all, two-dimensional layout of nearly anything can be a challenging problem, though especially difficult in a domain such as music engraving, where there is a well-developed, if not clearly formulated, standard practice. Rather than try to explicitly formulate the many aspects of good music notation, we seek to leverage existing examples, using these as a guide that will pull our results toward reasonable layout. This includes both the inter-symbol spacing as well as the categorical or discrete decisions of music notation (stem direction, beaming, etc.).

We have formulated the problem as a constrained optimization, specifically as quadratic programming, thus allowing us to incorporate many different objectives into our final result. Furthermore, the approach is fully automatic. It is worth noting, however, that no fully automatic approach will likely lead to the high quality notation we wish to produce. The optimization approach we present can easily be extended to handle explicit human guidance in the form of hand-specified constraints or additional terms for the quadratic objective function. When these are added, we still compute a global optimum that takes into account all other aspects of the objective function, as well as other constraints. Thus our approach provides a natural framework for including human-supplied guidance.

There are additional applications of the proposed techniques we have not presented within, but are still pursuing currently, which we hope will further support the overall framework we present. For instance, our symbolic representation of the music data easily allows for small transpositions in which notes are shifted a few staff positions up or down, while the transposed accidentals are handled in a formulaic (and correct) way. It is worth noting that this recipe for transposition runs into difficulty with larger transpositions that call for changes with stem directions and, perhaps, other notational choices.

Finally, we hope to soon pursue uses of notation designed to convey performance characteristics, rather than readability. For instance, we can present the horizontal position of each note as proportional to its actual onset time, thus allowing one to *see* the expressive use of timing, embedded in the notation itself. Additional variations would allow the visualization of fine-grained pitch, of interest for both tuning and vibrato, as well as dynamics. Our work with score alignment allows for estimation of the appropriate performance parameters, while the renotation ideas may prove useful for conveying this information in a way that is easy for the practicing musician to assimilate.

5. REFERENCES

[1] A. Hankinson, P. Roland, and I. Fujinaga, “The music encoding initiative as a document-encoding framework,” in *ISMIR*, 2011, pp. 293–298.

[2] M. Good, “Musicxml for notation and analysis,” *The virtual score: representation, retrieval, restoration*, vol. 12, pp. 113–124, 2001.

[3] L. Chen, R. Jin, and C. Raphael, “Renotation from optical music recognition,” in *International Conference on Mathematics and Computation in Music*, 2015, pp. 16–26.

[4] L. Pugin, R. Zitellini, and P. Roland, “Verovio: A library for engraving mei music notation into svg,” in *ISMIR*, 2014, pp. 107–112.

[5] H. W. Nienhuys and J. Nieuwenhuizen, “Lilypond, a system for automated music engraving,” in *Proceedings of the XIV Colloquium on Musical Informatics (XIV CIM 2003)*, 2003, pp. 167–72.

[6] C. Raphael and J. Wang, “New approaches to optical music recognition.” in *ISMIR*, 2011, pp. 305–310.

[7] R. Jin and C. Raphael, “Interpreting rhythm in optical music recognition.” in *ISMIR*, 2012, pp. 151–156.

[8] L. Chen, E. Stolterman, and C. Raphael, “Human-interactive optical music recognition,” in *ISMIR*, 2016, pp. 647–653.

[9] S. G. Kobourov, “Spring embedders and force directed graph drawing algorithms,” *arXiv preprint arXiv:1201.3011*, 2012.

[10] T. M. Fruchterman and E. M. Reingold, “Graph drawing by force-directed placement,” *Software: Practice and experience*, vol. 21, no. 11, pp. 1129–1164, 1991.

[11] P. Eades, “A heuristics for graph drawing,” *Congressus numerantium*, vol. 42, pp. 146–160, 1984.

[12] T. Kamada and S. Kawai, “An algorithm for drawing general undirected graphs,” *Information processing letters*, vol. 31, no. 1, pp. 7–15, 1989.

[13] R. K., “Algorithms and data structure for a music notation system based on guido notation,” Ph.D. dissertation, Technischen Universität Darmstadt, 2002.

[14] D. Bainbridge and T. Bell, “The challenge of optical music recognition,” *Computers and the Humanities*, vol. 35, no. 2, pp. 95–121, 2001.

[15] D. Bertsekas, *Nonlinear Programming*. Athena Scientific, 1995.

WALLACE: COMPOSING MUSIC FOR VARIABLE REVERBERATION

Filipe Lopes
Centro de Investigação em Psicologia da
Música e Educação Musical – INET-md
University of Aveiro
filipelopes@ua.pt

ABSTRACT

Reverberation is a sonic effect that has a profound impact in music. Its implications extend to many levels such as musical composition, musical performance, sound perception and, in fact, it nurtured the sonority of certain musical styles (e.g. plainchant). Such relationship was possible because the reverberation of concert halls is stable (i.e. does not drastically vary). However, what implications surface to music composition and music performance when reverberation is variable? How to compose and perform music for situations in which reverberation is constantly changing? This paper describes *Wallace*, a digital software application developed to make a given audio signal to flow across different impulse responses (IRs). Two pieces composed by the author using *Wallace* will be discussed and, lastly, some viewpoints about composing music for variable reverberation, particularly using *Wallace*, will be addressed.

1. INTRODUCTION

Up until the beginning of the twentieth century, engineers and architects knew little about reverberation, thus, could not plan in advance how a concert hall would sound after being built. In fact, the acoustical quality of many buildings designed for music was the result of pure chance [1]. That situation changed with Wallace Sabine (1868-1919), an engineer who is considered the father of architectural acoustics. Sabine discovered the mathematical relationship between size, materials of a room and its reverberation time [2]. His discovery revolutionised architectural acoustics.

Acoustic reverberation within closed spaces is more or less stable, which means that if conditions remain the same its acoustical qualities do not change drastically. This is why people are able to assign certain sonorities to certain generic spaces (e.g. cave, a bathroom or a hall). Additionally, the reason why some musical styles sound best in certain reverberation conditions (i.e. certain halls) is due to the fact that acoustic reverberation is stable [3]. Imagine a choir singing plainchant music in a cathedral or the same performance sang in a beach. The performance would sound quite different at both places but maybe would sound more appropriate at a cathedral. This is not only because we are accustomed to hear that kind of music in reverberant places but also because it flourished and matured in cathedrals and basilicas.

The advancements of analogue sound technologies during the twentieth century triggered many important investigations within the areas of music, sound and space. Apparatus such as loudspeakers and microphones had many implications for music composition and performance, particularly because it promoted new ideas to approach and work with space, thus, with reverberation. The piece *I'm sitting in a room* (1969) by Alvin Lucier represents an example of such experiments. A good historic introduction to how musicians and engineers employed efforts and imagination to develop mechanical machines and digital algorithms to simulate reverberation can be found here [4].

Nowadays, many halls are equipped with speakers, microphones and specific materials to overcome particular demands (e.g. voice intelligibility) or to make a given space to sound in a certain way. In fact, there are rooms that offer the possibility to change its reverberation characteristics by varying its wall panels (e.g. Espace de projection in IRCAM). Such possibility allows, for example, adapting reverberation time of the space to the demands of each piece during a concert.

Using computers one can compose music using simulations based on a "real" reverberation (i.e. the reverberation quality of a specific hall) but it is also possible to create imaginary reverberations. Regarding the first case, IR's are usually used in combination with convolution algorithms to simulate the sound of a specific sound source (e.g. voice) in a given space. This is useful, for example, to give musicians the impression of being at a specific place while they are being recorded in studio conditions [5]. Regarding the second case, one is able to develop reverberation algorithms that output a sonic result that does not derive from "the real world". These are only two examples about the use of computers to develop work focused/about reverberation. The work being done relating computers and reverberation is quite extensive and, nowadays, it has a big impact to simulate acoustic situations (i.e. auralization) as well as helping designing concert halls [6].

During musical performances, the acoustical reverberation quality is usually "static". This is very useful because reverberation dramatically affects our perception of the space but also musical performance (e.g. dynamics, tempo, rhythm, pitch, timbre). Composers compose music having in mind a given space (i.e. stable reverberation conditions) and it is based on that assumption that a composition can be performed in other spaces and still be faithful to the compositional ideas.

In electronic music composition, reverberation is frequently presented as a way to add depth (i.e. distance) to sounds. Although many different types of reverberation can be employed in a piece, as well as the possibility of automating its parameters using Digital Audio Workstations (DAWs), the author believes that the "compositional" approach to reverberation is usually passive. This means that reverberation is not frequently used as a composition building block but instead as an element to highlight other features or to "colour" the sound. For instance, the piece *Turenas* (1972), by John Chowning, employs an algorithm to change the reverberation according to intensity of the direct signal, however, it was devised to study the perception of distance and movement of sound using loudspeakers (i.e. localization and distance) [7]. Nonetheless, there are compositions that use variable reverberation in real time as a structural feature, such as in the piece *NoaNoa* (1992) by Kajia Saariaho. In this piece, reverberation is changing according to the intensity of the sound of the flute. According to the notes in the score: "The general idea here is: the quieter the sound, the longer the reverb".

2. WALLACE

2.1 Aim

Wallace (see Figure 1) is a software application to foster music compositions based on variable reverberation. Its implementation design is aimed at: 1) offering an easy way to make sound "travel" across independent reverberations automatically 2) exploring the relationship and implications between composition and performance of music in contexts in which reverberation is constantly changing.

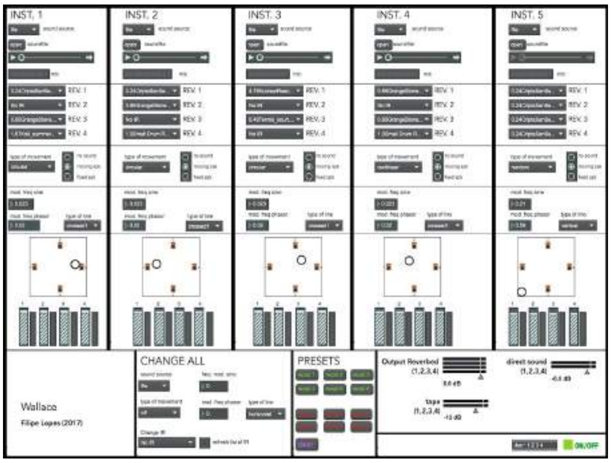


Figure 1. *Wallace*

2.2 Overview

Wallace makes a given audio signal to flow across different reverberations according to specific transition behaviours. Each sound signal from a given sound source is sent to a specific reverberation scheme (see Figure 2). This scheme performs real-time convolution with different IRs, thus, making the audio signal to flow across "different spaces". For each reverberation scheme, the user chooses four IRs from a default collection of IRs. The next step is to choose the transition type across the IRs. The final step consists on adjusting the gain level of each IR output.

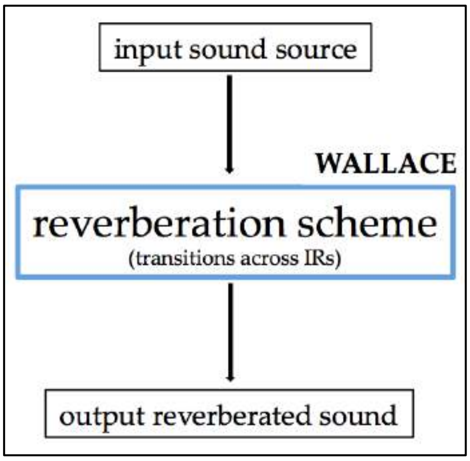


Figure 2. Audio signal flow in *Wallace*

2.2.1 Technical Info

Wallace was developed in MaxMSP [8]. By default, the total amount of possible sound sources is five. The sound sources can be sound files or live sound input (i.e. microphone). The convolution process is performed using the *HISSTools Impulse Response Toolbox* [9], particularly the MSP object *multiconvolve~*. This object performs real-time convolution.

The default IRs are included in an external folder named "IRs" and were retrieved from The Open Acoustic Impulse Response Library (Open AIR) [10]. The user, however, can add more IRs to the database and use them. *Wallace*, by default, is ready to output to a quadrophonic system, yet the user is able to choose a stereo output.

2.2.2 Transitions

The GUI (see Figure 3) defines a squared area to allow the user to visualize the transitions across the IRs. It features a small black circle that, according to its position, signals which IRs is being "used" (i.e. heard) to process the audio signal. The closer the circle is to a "speaker" within the defined squared area (i.e. specific number as displayed in the GUI), the louder that specific convolution process (i.e. sound result) is heard. If the circle is in the middle of the squared area, however, the sound result will be a mixture of all the sound convolutions.

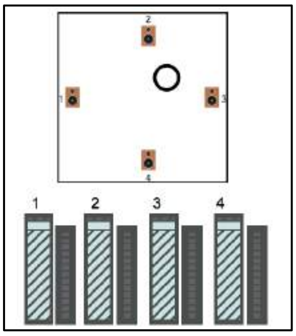
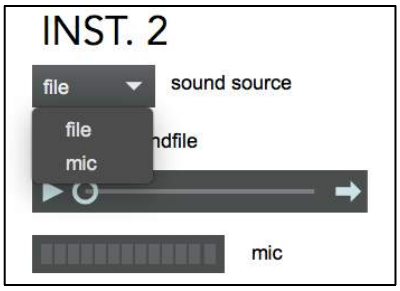


Figure 3.

There are a total of four transitions options to manage each reverberation scheme: "off", "circular", "rectilinear" and "random". The option "off" means that there is no automatic transition, thus, the sound result is either stable (i.e. same reverberation, thus circle stationary) or changing according to the user (i.e. user moves the circle as desired using the mouse). The option "circular" makes the circle to move in a circular fashion modulated by a sine wave generator. The user defines the rate of that movement by increasing or decreasing the frequency fed to a cycle~ MSP object. The option "rectilinear" makes the circle to move in a rectilinear fashion between two extremes (e.g. 1 and 3). The rectilinear movement can be horizontal, vertical or crossed. The sound result at the extremes is the result of one convolution process whereas in the middle the sound result consists of a mixture of all the convolutions. Once again, the user defines the rate of that movement by increasing or decreasing the frequency fed to a phasor~ MSP object. Finally, the option "random" performs a random choice within the defined squared area and smoothly moves the circle to that spot. This procedure is repeated until furthermore instructions.

2.3 Workflow

The first step is to choose one of two possible sound sources: *file* or *mic.* The first case refers to a sound file while the second refers to sound input (see Figure 4).

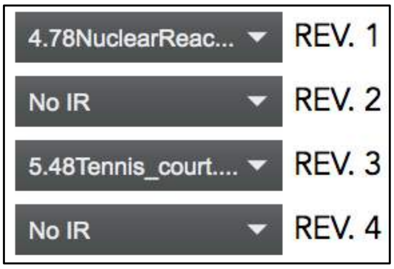


INST. 2

file sound source

Figure 4. Input options

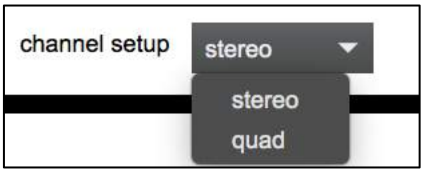
The second step consists in choosing the four different IRs to be used in the reverberation scheme. One is able to choose the option "No IR", which means dry sound (i.e. sound not flowing to the convolution process) (see Figure 5).



4.78NuclearReac... REV. 1
No IR REV. 2
5.48Tennis_court.... REV. 3
No IR REV. 4

Figure 5. IRs chosen for a reverberation scheme

The third step consists on choosing the type of transition behaviour (see 2.2.2.). Finally, one needs to decide if the reverberation scheme is delivered to four speakers (i.e. one IR for each speaker) or two speakers (i.e. two IRs for each speaker) (see Figure 6).



channel setup stereo

Figure 6. Sound output options

3. VARIAÇÕES SOBRE ESPAÇO

3.1 Overview

Two different pieces were composed to experiment composing music based on variable reverberation. I will now describe those pieces, particularly focusing on the impressions I collected while composing the pieces and listening to them in concert situations.

3.2 Variações sobre Espaço #1

The first piece is for soprano saxophone and live electronics only using *Wallace*. For the saxophone part I deliberately used musical phrases containing contrasting elements (e.g. high pitched sounds vs low pitched sounds, forte vs piano, sound vs silence). Although the formal structure of the saxophone score is linear (i.e. not open form), it is composed of many sequenced small musical gestures to enhance the contrasting elements (see Figure 7).

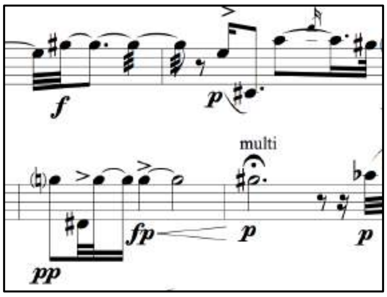


Figure 7. Excerpt of the saxophone score

The decision to compose the saxophone score based on contrasting elements originated from previous sessions with the saxophonist. During those experimental sessions, it seemed that using sharp and contrasting sound gestures helped to "perform" and hear the reverberation nuances. With the inclusion of that kind of phrasing texture in the final score, I wanted to 1) listen to its sound results using *Wallace* 2) assess if I could experience the saxophone sound and the reverberation(s) "in dialogue" (e.g. overlapping sounds and pitches).

During one of the rehearsals, by chance, I discovered that using the sound of the human voice, in comparison with the sound of the saxophone for the same reverberation scheme, I perceived much more clearly the different reverberations. I believe that this might be related to 1) the spoken human voice is noisier (i.e. irregular spectrum) when compared to the saxophone sound (i.e. "regular" spectrum), thus, more radical spectral changes occur when the audio is being processed and 2) the human voice is hardwired to our daily experience of the world, thus, the humans instantly recognize the smallest nuances.

This piece was performed in May 2016 and can be heard here [11].

3.3 Variações sobre Espaço #2

This piece is for quintet (flute, clarinet, piano, violin and violoncello), tape and live electronics (*Wallace*). It comprises 5 movements and the tape is comprised of distinctive soundscapes (e.g. forest, inside a church, in the countryside, late night at a small village).

This instrumental setup allowed me to explore different facets of mixing/blending reverberations (i.e. IRs) when compared with the strategies I used in *Variações sobre Espaço #1*. Such facets include 1) the articulation of the ensemble (i.e. several sound sources) with *Wallace* and its consequence on the overall sound result 2) repeat pitched notes at a given pulse (e.g. repeated quarter notes) to hear its sonic nuances changing across the different reverberations 3) use contrasting instrumental textures (e.g. tutti vs solo). Some of the questions I asked myself were: Will I hear many spaces blending with each other? Will I hear different instruments in different spaces? Will I hear just a single reverberation composed of several reverberations?

The instrumental texture/notation used in the beginning of the piece is very common sounding, however, with each movement, some musical aspects are "frozen" or "simplified" (e.g. note duration, the rate at which harmony changes, textures, dynamics, rhythm). In the last movement there is only a continuous and spacy melody played by the piano, only punctuated by slight gestures played by the remaining instruments. The reason I composed like that was to be able to experiment and listen to many sonic textures (e.g. complex vs simple) between the sound produced by the instruments and *Wallace*.

During rehearsals, I discovered that I could listen and perceive many simultaneous different reverberations when there were soundscape sounds permanently playing in the background. Consequently, I recorded several and distinct soundscapes in order to build a database of "background soundscapes". Most importantly, these soundscapes are not meant to stood out but instead install a quiet "background space".

This piece was performed twice. Each performance occurred at a different concert hall and each concert hall had contrasting natural reverberation qualities. The first concert hall had a pronounced natural reverberation (~ 2 seconds) and the second one had little natural reverberation. The experience of listening to the piece, particularly experiencing the transitions between different virtual spaces (i.e. IRs), felt quite different in each concert hall. The first two movements (i.e. more complex rhythmical textures) seemed more interesting when performed in the concert hall with little reverberation whereas the last movements (i.e. simpler rhythmical textures) felt more appropriate for the concert hall with noticeable natural reverberation.

This piece was performed in November 2016 and can be heard here [12].

4. CONCLUSIONS

The main purpose of this research is to devise ideas about composing music for variable reverberation. *Wallace* is a resource to pursue such compositional intents. It is a digital software application developed to make a given audio signal to flow by different reverberations. In addition, *Wallace* offers possibilities to make automatic transitions between the different reverberations.

The practical compositional work has led me to some conclusions, specifically: the spoken human voice is the sound that, from the stand point of sound perception, best illustrates different reverberations; the natural reverberation of each space plays a decisive role in the perceived sound output produced by *Wallace*, thus, each performance will not sound the same in concert halls with different reverberations. This might mean that there is no perfect concert hall to use *Wallace*, instead, each composition will be (and should be composed to be) in dialogue with the real and virtual space; lastly, the use of continuous quiet soundscape sounds imprints a sense of background space which helps *Wallace's* sound output to stand out.

During the course of composing the aforementioned pieces, as well as implementing *Wallace*, I established a generic compositional approach (see Figure 8). It defines three main ideas to be addressed while composing music for variable reverberation, particularly using *Wallace*. The first idea suggests one to think about the balance between dry sound/open space/soundscape vs wet sound/closed space/reverberation; the second idea suggests one to think about the balance between static virtual reverberation vs variable virtual reverberation; the third idea, in the case of using variable reverberation schemes, suggests one to think about how to employ transitions between distinct reverberations. In addition, the balances aforementioned don't (should not) have to be constant during the course of a piece. Instead, it seems to me interesting to shift the balances during the performance of the piece.

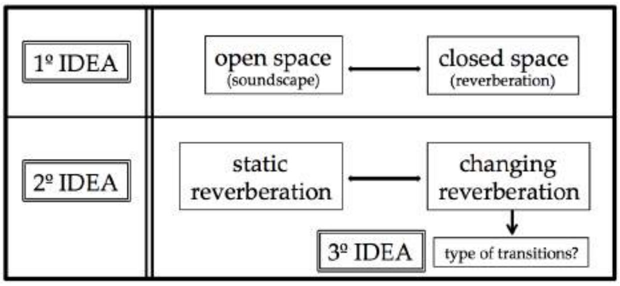


Figure 8. Generic compositional topics to consider when composing music for variable reverberation

5. FUTURE WORK

Future work includes the composition of new pieces, the inclusion of more IRs in the default database, the elaboration of documentation (e.g. video tutorials and performance videos) and the design of new models to make IR transitions within each reverberation scheme (e.g. more automatic transition movements, analyze the input sound of a sound source and map a specific audio feature to move the black circle). Furthermore, the GUI is going to be redesign and *Wallace* application and source code are going to be released very soon at filipelopes.net.

Acknowledgments

Many thanks to CIPEM- Inet-MD, Henrique Portovedo, Miguel Azguime, Miso Music Portugal, Sond'Ar-Te Ensemble, João Vidinha and Rui Penha.

6. REFERENCES

[1] M. Forsyth, Buildings for Music. Cambridge, MA: MIT Press, 1985

[2] L. L. Henrique, Acústica Musical. Edição 5ª. Lisboa, Fundação Calouste Gulbenkian.

[3] A. P. de O. Carvalho, Acústica Ambiental. Ed. 84. Faculdade de Engenharia da Universidade do Porto, 2013

[4] B. Blesser "An interdisciplinary synthesis of reverberation viewpoints" in Journal of the Audio Engineering Society, 2001,49(10), pp. 867-903

[5] B. Pentcheva "Hagia Sophia and Multisensory Aesthetics" in Gesta, 2011, 50(2), pp. 93-111

[6] https://www.wired.com/2017/01/happens-algorithms-design-concert-hall-stunning-elbphilharmonie/

[7] Chowning, J. (2011). Turenas: the realization of a dream. Communication presented at the Journées d'Informatique Musicale 2011, Saint-Etienne

[8] http://www.cycling74.com

[9] A. Harker, P. Alexandre Tremblay, "The HISSTools impulse response toolbox: Convolution for the masses." in Proc. Int. Computer Music Conf, 2012, pp. 148-155.

[10] http://www.openairlib.net

[11] https://youtu.be/XnYBrRhMDhg?t=32m46s

[12] https://soundcloud.com/filklopes/variacoes-sobre-espaco-2-mov-1-ato-3

REAL-TIME PHYSICAL MODEL FOR SYNTHESIS OF SWORD SWING SOUNDS

Rod Selfridge
Media and Arts Technology
Doctoral College,
r.selfridge@qmul.ac.uk

David Moffat
Centre for Digital Music
EECS Dept.,
Queen Mary University of London,
Mile End Road, UK E1 4NS
d.j.moffat@qmul.ac.uk

Joshua D Reiss
Centre for Digital Music
joshua.reiss@qmul.ac.uk

ABSTRACT

Sword sounds are synthesised by physical models in real-time. A number of compact sound sources are used along the length of the sword which replicate the swoosh sound when swung through the air. Listening tests are carried out which reveal a model with reduced physics is perceived as more authentic. The model is further developed to be controlled by a Wii Controller and successfully extended to include sounds of a baseball bat and golf club.

1. INTRODUCTION AND RELATED WORK

A number of models have been created to synthesise the sound of a sword sweeping through the air. Commonly these are based on one of two methods - signal based and physical models [1].

The advantage of a signal based model is that it is computationally inexpensive to replicate the spectrum of a sound using established techniques such as additive synthesis or noise shaping. A drawback of this approach could be relating changes in signal properties to the physical processes creating the sound. For example, an increase in speed of a sword not only changes the fundamental tone frequency but also the gain. Therefore changing one signal based property could lose realism in another.

Physical models aim to replicate the physics behind the sound generation process. Sounds generated by these models have the advantage of having greater accuracy in relation to the parameter settings but a drawback can be that the computational cost required to produce sounds is often high. Also, it is often the case that the sound effects cannot be adapted quickly to parameter adjustments and therefore not able to operate in real-time.

In the middle of these traditional techniques lie physically inspired models. This hybrid approach replicates the signal a sound produces but adds, to a greater or lesser extent, characteristics of the physics that are behind the sound creation. For a simple sword model this might be noise shaping with a bandpass filter with centre frequency proportional to the speed of the swing. A variety of examples of physically inspired models were given in [2]; the model for whistling wires being exactly the bandpass filter mentioned.

Figure 1. Computer generated character with a sword. Motion and dimension parameters can be determined by the game engine. (https://www.youtube.com/watch?v=zVvNthqKQIk)

Four different sword models were evaluated in [3]. Here the application was for interactive gaming and the evaluation was focused on perception and preference rather than accuracy of sound. The user was able to interact with the sound effect through the use of a Wii Controller. One model was a band filtered noise signal with the centre frequency proportional to the acceleration of the controller. A physically inspired model replicated the dominant frequency modes extracted from a recording of a bamboo stick swung through the air. The amplitude of the model was mapped to the real-time acceleration data.

The other synthesis methods both mapped acceleration data from the Wii Controller to different parameters; one using the data to threshold between two audio samples, the other a granular synthesis method mapping acceleration to the playback speed of grains. Tests revealed the granular synthesis was the preferred method for expression and perception. One possible reason that the physical model was less popular could be the lack of correlation between speed and frequency pitch, which the band filtered noise had. This may also be present in the granular model.

A signal based approach to a variety of environmental sound effects, including sword whoosh, waves and wind sounds was undertaken in [1]. Analysis and synthesis occur in the frequency domain using a sub-band method to produce coloured noise. In [4] a rapier sword sound was replicated but this focused on the impact rather than the swoosh when swung through the air.

A physical model of swords sounds was explored in [5]. Here offline sound textures are generated based on the physical dimensions of the sword. The sound textures are then played back with speed proportional to the movement. The sound textures are generated using computational fluid dynamics software, (CFD), solving the Navier Stokes equations and using Lighthill's acoustic analogy [6] extended by Curle's Method [7]. In this model, [5], the sword is split into a number of *compact sources*, (discussed in section 2), spaced along the length of the sword. As a sword was swept thought the air each source will move at a different speed and the sound texture for that source adjusted accordingly. The sound from each source was summed and output to the listener.

A Japanese Katana sword was analysed in [8] from wind tunnel experiments. A number of harmonics from vortex shedding were observed along with additional harmonics from a cavity tone due to the *shinogi* or blood grooves in the profile of the sword.

This paper presents the design, implementation and analysis of a real-time physical model that can be used to produce sounds similar to those of a sword swooshing through the air. It builds on previous work by the authors to create a real-time physical model of an Aeolian tone [9].

Our model offers the user control over parameters such as the arc length of the swing, sweep angles, top speed of the swing, dimensions of the blade as well as calculating what a listener will hear from a given observation point. The parameters available give the user the ability to model a wide variety of sword profiles, as well as other objects that produce similar sounds.

In addition a version of the model has been implemented with control mapped to the movement of a Wii controller. It is envisaged that many of the parameters available to the user could be set by a game engine and therefore the sounds generated directly. This would allow a correlation of the sound generated to the movement and weapon being used by a character, (Fig. 1).

2. THEORY

The fundamental aeroacoustic sound created when a cylinder is swung through the air is the Aeolian tone. This sound is generated when fluid passes around the cylinder and vortices are shed from opposite sides. This causes oscillating lift and drag forces which in turn produce tones of different strength and propagation directions.

A brief overview of the Aeolian tone will be given here, including some fundamental equations. For greater depth the reader is directed to [9].

Harmonic	Frequency	Gain
Drag dipole fundamental (f_d)	$2f_l(t)$	$0.1\overline{I_l(t)}$
Lift dipole 1st harmonic	$3f_l(t)$	$0.6\overline{I_l(t)}$
Drag dipole 1st harmonic	$4f_l(t)$	$0.0125\overline{I_l(t)}$
Lift dipole 2nd harmonic	$5f_l(t)$	$0.1\overline{I_l(t)}$

Table 1. Additional frequencies and gains as a function of the lift dipole fundamental frequency$f_l(t)$.

2.1 Tone Frequency

Strouhal (1878) defined a useful relationship between the tone frequency f_l air speed u and cylinder diameter d (Eqn. (1)). The variable S_t is known as the Strouhal number.

$$S_t(t) = \frac{f_l(t)\,u(t)}{d} \tag{1}$$

As air flows around cylinder, vortices are shed causing a fluctuating lift force normal to the flow dominated by the fundamental frequency, f_l. Simultaneously a fluctuating drag force is present with frequency, f_d, twice that of the lift frequency. The drag acts in line with the the air flow and it was noted in [10] that, "*The amplitude of the fluctuating lift is approximately ten times greater than that of the fluctuating drag.*"

It was shown in [7] and confirmed in [11] that aeroacoustic sounds, in low flow speed situations, could be modelled by the summation of *compact sources*, namely monopoles, dipoles and quadrupoles. Aeolian tones can be represented by dipole sources, one for the lift frequency and one for the drag; each source includes a number of harmonics.

The turbulence around the cylinder affects the frequency of the tones produced. A measure of this turbulence is given by a dimensionless variable, the *Reynolds number*, R_e, given by the relationship in Eqn. (2).

$$R_e(t) = \frac{\rho_{air}\,d\,.u(t)}{\mu_{air}} \tag{2}$$

where ρ_{air} and μ_{air} are the density and viscosity of air respectively. The value of the Strouhal number has been found to be related to the Reynolds number. An experimental study of this relationship was performed in [12], giving the following equation:

$$S_t(t) = \lambda + \frac{\tau}{\sqrt{R_e(t)}} \tag{3}$$

where λ and τ are constants and given in Table 1 of [12], (additional values are calculated in [9]). The different values represent the turbulence regions of the flow, starting at laminar up to sub-critical.

With the Strouhal number obtained, diameter and air speed known, we can apply them to Eqn. (1) and obtain the fundamental frequency, $f_l(t)$, of the aeolian tone, generated by the lift force. Once $f_l(t)$ has been calculated, we can calculate the drag dipole frequency and harmonics as shown in Table 1.

Each frequency peak has a bandwidth which is proportional to the Reynolds number. Equations to calculate the bandwidth are given in [9], derived from data published in [13].

2.2 Source Gain

Once the fundamental frequency has been calculated the source intensity and propagation pattern can be calculated. The time-averaged Acoustic Intensity $\overline{I_l(t)}$ (W/m^2), of an Aeolian tone lift dipole source and the time-averaging period are given in [14]. The time-averaged Acoustic Intensity is given as:

$$\overline{I_l(t)} \approx \frac{\sqrt{2\pi}\kappa^2 S_t(t)^2 l\, b\, \rho\, u(t)^6 \sin^2\theta \cos^2\varphi}{32c^3r^2(1-M(t)\cos\theta)^4} \qquad (4)$$

where $M(t)$ is the Mach number; $u(t)/c$, where c is the speed of sound. The elevation angle, azimuth angle and distance between listener and source are given by θ, φ, and r respectively. The constant κ is set to 1. The correlation length, l, given as a multiple of diameters, indicates the span-wise length that the vortex shedding is in phase; after this the vortices become decorrelated. $\overline{I_l(t)}$ is applied to the fundamental frequency and scaled for the harmonics as shown in Table 1. The gain for the drag dipole is obtained from [10] and the lift dipole harmonics values from [15]. The drag dipole 1st harmonic was witnessed in computational simulations and added at the appropriate value.

2.3 Wake Noise

As the Reynolds number increases, the vortices diffuse rapidly and merge into a turbulent wake. The wake produces wide band noise modelled by lateral quadrupole sources whose intensities vary with $u(t)^8$ [16].

It is noted in [16] that there is very little noise content below the lift dipole fundamental frequency. Further, it has been shown in [17] that the roll off of the amplitude of the turbulent noise, above the fundamental frequency is $f_l(t)^{-2}$. An equation for calculating the wake noise is given in [9].

3. IMPLEMENTATION

Implementation of each individual compact source is given in [9]. The basic concept of the sword model is to line up a number of the compact sources to recreate the sound created as a blade swings through the air.

Like the original compact source model presented in [9], this model was developed using the Pure Data programming platform. The intensity given in Eqn. (4) is time averaged which caused an issue for this model due to the swing time being shorter than the averaging process. In the case of this model the intensity was implemented as an instantaneous value.

Our model has been given a simple interface to allow the user to manipulate physical properties such as sword thickness, length, top speed at the tip and sweep angles. Some or all of these parameters can be directly mapped to graphics and animation within a game environment.

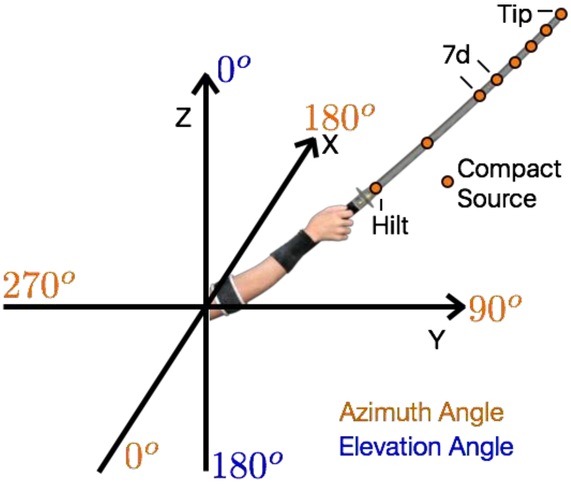


Figure 2. Position of 8 compact sources and coordinates used in sword model.

For ease of design the thickness down the blade of the sword was taken as a linear interpolation between the thickness set at the hilt and that set at the tip. The diameter of an individual source is then set as the thickness corresponding to its position on the blade. Equation 4 shows that the gain is proportional to $u(t)^6$. As a sword arcs, the blade moves faster through the air at the tip than at the hilt. Hence the sources nearer the tip will have a greater influence over the sound generated than those nearer the hilt.

Eight compact sources were used to model the sword in our model. Two of the sources were fixed at the tip and the hilt. Five sources were then positioned back from the tip at a spacing of 7 times the diameter, d. The final source was placed equidistant between the last one from the group at the tip and the hilt. This is illustrated in Fig. 2.

This positioning of the six sources at the tip was equivalent to each source having a set correlation length of $7d$, (see Section 2.2). A range of correlation values from $17d$ to $3d$ are given in [18] depending on Reynolds number. A plot showing similar values is shown in [19]. [1]

In this model, the position of the sources has to be chosen a priori to the calculation of the Reynolds number; the value $7d$ was chosen as a compromise, covering a reasonable length of the sword for a wide range of speeds (hence Reynolds numbers). For illustration, examining a sword of length 1.117 metres long and diameter ranging from 0.013 metres at the hilt to 0.008 metres at the tip. The 8 sources positioned as described equates to a coverage of 43% of the blade length being represented by the sources. As shown in Fig. 2 the positioning of sources capture the area which contributes most to the sound.

The coordinate system used for the model is shown in Fig. 2. The centre of the sword arc is at the origin and each sweep is set as circular. The distance travelled by each source during an arc is a great circle on a sphere with a radius equal to that of the source.

The user specifies the sweep of the sword by setting a start and finish for azimuth and elevation angles as well as

[1] In [5] the correlation length of $3d$ was used; the number of sources set to match the length of the sword.

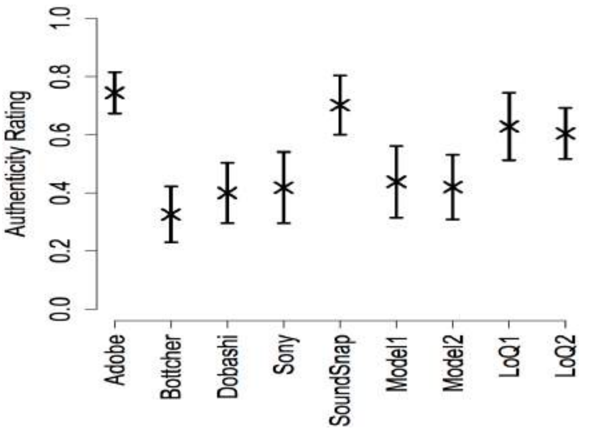


Figure 3. Sword sounds clips rated by authenticity. (Mean and 95% confidence shown.)

the top speed of the tip. It is presumed the sword starts and finishes at rest and also uniform acceleration and deceleration; the average speed is $top\ speed/2$. The blade reaches top speed when it crosses the xy plane, an azimuth angle of 180°. Any graphical character swinging the sword can be oriented in any direction and the elevation and azimuth adjusted accordingly. Knowledge of the distance travelled and average speed enables us to calculate the ramp time up to top speed as well as back to rest for each source.

We know that the sword sweep is a 2 dimensional plane in a 3 dimensional environment. The observer is taken to be a point in that environment. To calculate the elevation and azimuth of each source to observer a projection matrix is created to find the position in the plane of the sword perpendicular to the observer. The angles can be calculated using trigonometry identities as can the overall distance between the observer and individual sources.

Panning is included as the sound moves across the xy plane as well as the Doppler Effect. It was shown in [20] that the addition of the Doppler Effect increases the natural perception. This effect is taken into account when the sword is moving towards or away from the observer and frequencies adjusted accordingly.

4. RESULTS AND ANALYSIS

4.1 Listening Test

A listening test was carried out using [21], asking participants to rate a number of sounds for authenticity. Table 2 classifies the sounds used for the test; all sounds were wav files. The sample duration was calculated as the total length of the sample, where the loudness was greater than -90 dBFS. A total of 25 participants took part, aged from 16 to 77 years old, with a mean age of 39.1 years and standard deviation of 16.8, (one participant declined to state their age). The sounds chosen included 3 samples from effects libraries, a recording of the synthesis model by [3], as well as one from [5]. Users were provided with an interface that allowed them to play back each sound sample and rate the sounds on a scale. Participants auditioned the sounds through a pair of AKG K550 Headphones - this was done to remove any natural acoustic impact of the sounds being played in any environment. The users were asked to rate the sounds in terms of "authenticity". Sample order was randomised and all samples were loudness normalised[2]. Participants were allowed to audition any sample as many times as they wanted, and on average, participants played each sound 8 times.

Source	Classification	Duration (ms)
Adobe	Sample	481.7
Bottcher	Granular / Additive mix	218.6
Dobashi	CFD	496.7
Sony	Sample	1138.4
SoundSnap	Sample	161.0
Model 1	Physical Model	917.1
Model 2	Physical Model	928.8
LoQ 1	Physical Model	731.1
LoQ 2	Physical Model	1079.7

Table 2. Clips used for listening test.

Two different sword sounds from our model were used, one thin and long and one thicker and slightly shorter (Model 1 and Model 2). Lower quality versions of the two swords sounds were also generated, keeping all the dimensions identical, (LoQ1 and LoQ2). The lower quality versions use a model with reduced knowledge of the physics with a fixed the Strouhal number St at 0.2, removed the wake noise source and fixed the bandwidth of the fundamental and harmonics, all filters Q value set to 10.

The mean authenticity rating plus 95% confidence intervals (approximately two standard deviations) is shown in Fig 3. The sampled sounds taken from the Adobe and Soundsnap libraries are clearly rated the highest. This is not surprising as it is possible every effort has been spent mastering the sounds to achieve the best effects. The sound rated the worst was the one taken from [3]. It is know that this sound was not generated for authenticity, rather perception and preference.

For a deeper analysis we performed a one-way ANOVA to determine the impact of synthesis method on the user authenticity ratings, and a significant effect is identified (F(8,216) = 0.62, $p<0.00001$). A Tukey post-hoc test was performed to determine the comparison between each individual synthesis method, and the results are presented in Table 3.

It can be seen from Table 3, that all synthesis methods apart from the two low quality methods are significantly different from the recorded samples. The low quality synthesis methods are both considered as indistinguishable from the recorded sound samples. It should also be noted that the low quality synthesis is not significantly different from any many synthesis methods. As such, these synthesis methods did not significantly out performed any other method, other than Bottcher.

Analysis shows that the sounds generated by our sword model has a perceptual rating on par with the Sony recorded sample as well as a granular / additive synthesis

[2] in compliance with ITU-R BS.1534-1

	Adobe	Bottcher	Dobashi	Sony	SoundSnap	Model 1	Model 2	LoQ 1	LoQ 2
Adobe	.	****	***	***	-	***	***	-	-
Bottcher	****	.	-	-	****	-	-	**	*
Dobashi	***	-	.	-	***	-	-	-	-
Sony	***	-	-	.	**	-	-	-	-
SoundSnap	-	****	***	**	.	**	***	-	-
Model 1	***	-	-	-	**	.	-	-	-
Model 2	***	-	-	-	***	-	.	-	-
LoQ 1	-	**	-	-	-	-	-	.	-
LoQ 2	-	*	-	-	-	-	-	-	.

Table 3. Results of a Tukey post hoc test. **** = p<0.0001, *** = p<0.001, ** = p<0.01, * = p<0.05, - = p> 0.05

technique [3] and an offline CFD technique [5]. The advantage of our model being the ability to relate the output sound to parameters of a physical sword swinging through the air and its real-time operation.

The surprise in the listening test is the results for the low quality model. Although they did not significantly out perform any other method they were also considered perceptually indistinguishable from the top performing recorded samples. One reason could be due to the fixed bandwidth which give the impression of more turbulence as the sword sweeps; the full models sound thinner in comparison. It should also be recognised that the full model is intended to replicate the physics as accurately as possible, as would be produced in an anechoic chamber.

Although there is no conclusive evidence, it is felt that fixing the Strouhal number at 0.2 and removal of the wake sound within each source will have a lower perceptual effect than fixing the bandwidth. Computationally this saves a number of calculations which was one of our goals when developing the model. It is possible that removing one of the lift harmonics and the drag harmonic will also have little effect on the perception of authenticity for this model but again save on computations.

It is reasonable to presume that participants could be more familiar with sword sound effects than the actual sounds generated when physically using a sword. Participants were offered the chance to comment on any sound file they wished. A comment on Model 1 was *"seems to be a very small object, (almost 'too small')"* and on the low quality version of the same sword, LoQ1, is *"a very plausible light weight sword"*. Similarly for Model 2 a comment was *"the frequencies constituting the sound seem awkward / implausible"* and for equivalent the low quality model, LoQ2, *"good sound"*, *"arteficial"* and' *"sounds model like scrapping fabric"*.

4.2 Interactive Test

To allow participants to interact with the model a Wii Controller was linked to a MacBook via bluetooth and a free trial version of the commercial software, OSCulator [3] . The angular velocity values from the gyroscope in the Wii controller are mapped to the speed of the source at the hilt and scaled depending on radius of the remaining sources.

Figure 4. Test participant swinging the Wii Controller synthesising a golf swing sound.

It was decided to fix the listener position in this model since the elevation and the azimuth values obtained when moving the Wii Controller drifted, which would created confusing pan values when swinging.

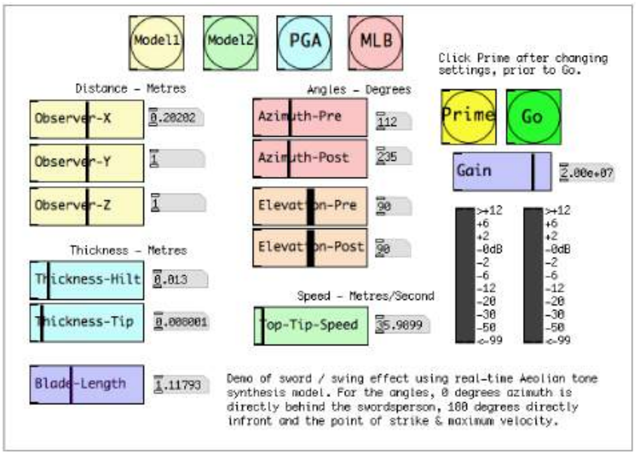
The start model given to users was equivalent to a broom handle of 1.5 metres long and 0.024 metres thick. This allowed the participant to get used to the model, including how fast they had to swing to get the results required. The gain was always set low but increased if requested.

Once participants were familiar with the broom handle model, four other presets were demonstrated including the two swords that were within in the listening test. We extended the sound effects that may be generated by our model by adding in the dimensions of a baseball bat and that a golf club. Fig. 4 shows a participant interacting with the golf club synthesis model.

Once participants had interacted with the presets to their satisfaction they were invited to change the dimensions of the model to their desire. Blade thickness and length, as well as gain, could be varied by the participant by using the buttons on the Wii controller. More often than not, participants who wished to change the parameters were guided by the author. The usual request was to make a sword in the region of 3 metres long, more than likely too heavy and cumbersome in reality but fun to interact with. The sword demo GUI is shown in Fig. 5.

One participant asked the author to replicate the sound of a pickaxe and another a martial arts Bo staff. Both stated they were satisfied with the results. Once they had finished interacting with the models, participants were asked to fill

[3] https://osculator.net



Click Prime after changing settings, prior to Go.

Observer-X 0.20202

Observer-Y 1

Observer-Z 1

Azimuth-Pre 112

Azimuth-Post 235

Elevation-Pre 90

Elevation-Post 90

Gain 2.00e+07

Thickness-Hilt 0.013

Thickness-Tip 0.008001

Top-Tip-Speed 35.9899

Blade-Length 1.11793

Demo of sword / swing effect using real-time Aeolian tone synthesis model. For the angles, 0 degrees azimuth is directly behind the swordsperson, 180 degrees directly infront and the point of strike & maximum velocity.

Figure 5. Pure Data GUI for the sword, golf club and baseball bat sounds.

	Mean Rating (Standard Deviation)
The experience was enjoyable.	4.46 (0.65)
I could achieve the sounds I wanted.	4.08 (0.63)
The sounds correlated with my movements.	4.38 (0.75)
I found the effect and interface easy to use.	4.08 (0.98)
I would use these sounds in a game.	3.92 (1.16)
Please rate the quality of the sounds.	4.19 (0.75)

Table 4. Ratings for Wii Controller interaction. (Ratings from *strongly disagree = 1* to *strongly agree = 5*, except the last question, *very poor = 1* to *excellent = 5*.)

in a short questionnaire, stating how much they agreed with set statements, shown in Table 4.

Results show that participants overwhelmingly enjoyed interacting with the models. There is also a very high result for the correlation between participants' movements and the sounds generated, a fundamental goal of the physical model. A number of comments were recorded after the interactive tests which included *"Fun experiment, the sounds drove me on and created a sense of excitement"*, *"..it's the lack of weight on the tip.."* and *"The parametrisation of the sound model made a lot of sense"*.[4]

5. CONCLUSION

The swoosh sound generated from a sword as it is swung through the air is successfully replicated by an implementation of a number of compact source models given in [9]. Vortex shedding frequencies are generated in real-time while calculating intensity, bandwidths and harmonics.

Listening tests show model with reduced physics is perceived as more authentic than the model with higher physical knowledge. The sounds of Model 1 and Model 2 perceived as just as authentic as other synthesis models; sample sounds were rated highest for authenticity out of all the sounds.

The model was successfully adapted to allow interaction with a Wii Controller and extended to include sounds of a baseball bat and golf club, as well as various gigantic swords and a pickaxe. Participants found this experience fun and engaging but also highlights the versatility of the model to synthesise bats and clubs. The range of sound effects this model is able to synthesise is an area for future research.

The linking of the sword model to a game engine or virtual reality environment would be of great value. This would give users the ability to visualise actions increasing the correlation between sound and movement.

It will be of value to investigate the balance between more compact sources and reducing the physics included in the compact source model. In contrast, adding the natural vibration frequency of the sword may have an effect, especially if the shedding frequency and the natural frequency match. Including cavity tones as witnessed by the wind tunnel experiments in [8], will increase the variety of profiles that can be modelled.

Acknowledgments

Supported by EPSRC, grant number EP/G03723X/1.

6. REFERENCES

[1] D. Marelli *et al.*, "Time-frequency synthesis of noisy sounds with narrow spectral components," *IEEE Transactions on Audio, Speech and Language Processing*, 2010.

[2] A. Farnell, *Designing sound.* MIT Press Cambridge, 2010.

[3] N. Böttcher and S. Serafin, "Design and evaluation of physically inspired models of sound effects in computer games," in *Audio Engineering Society Conference: 35th International Conference: Audio for Games*, 2009.

[4] L. Mengual, D. Moffat, and J. D. Reiss, "Modal synthesis of weapon sounds," in *Audio Engineering Society Conference: 61st International Conference: Audio for Games*, 2016.

[5] Y. Dobashi, T. Yamamoto, and T. Nishita, "Real-time rendering of aerodynamic sound using sound textures based on computational fluid dynamics," in *ACM Transactions on Graphics*, 2003.

[6] M. J. Lighthill, "On sound generated aerodynamically. i. general theory," *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 1952.

[7] N. Curle, "The influence of solid boundaries upon aerodynamic sound," in *Proceedings of the Royal So-*

[4] A copy of all the sound files used in the perceptual evaluation test as well as a copy of the PureData sword demo model can be obtained from https://code.soundsoftware.ac.uk/projects/physical-model-of-a-sword-sound.

ciety of London A: Mathematical, Physical and Engineering Sciences, 1955.

[8] M. Roger, “Coupled oscillations in the aeroacoustics of a katana blade,” *Journal of the Acoustical Society of America*, 2008.

[9] R. Selfridge *et al.*, “Physically derived synthesis model of Aeolian tones,” *Best Student Paper Award - 141st Audio Engineering Society Convention USA*, 2016.

[10] C. Cheong *et al.*, “Computation of Aeolian tone from a circular cylinder using source models,” *Applied Acoustics*, 2008.

[11] J. Gerrard, “Measurements of the sound from circular cylinders in an air stream,” *Proceedings of the Physical Society. Section B*, 1955.

[12] U. Fey, M. König, and H. Eckelmann, “A new Strouhal-Reynolds-number relationship for the circular cylinder in the range 47 105,” *Physics of Fluids*, 1998.

[13] C. Norberg, “Effects of Reynolds number and a low-intensity freestream turbulence on the flow around a circular cylinder,” *Chalmers University, Goteborg, Sweden, Technological Publications*, 1987.

[14] M. E. Goldstein, “Aeroacoustics,” *New York, McGraw-Hill International Book Co., 1976.*, 1976.

[15] J. C. Hardin and S. L. Lamkin, “Aeroacoustic computation of cylinder wake flow.” *AIAA journal*, 1984.

[16] B. Etkin, G. K. Korbacher, and R. T. Keefe, “Acoustic radiation from a stationary cylinder in a fluid stream (Aeolian tones),” *Journal of the Acoustical Society of America*, 1957.

[17] A. Powell, “Similarity and turbulent jet noise,” *Journal of the Acoustical Society of America*, 1959.

[18] O. M. Phillips, “The intensity of Aeolian tones,” *Journal of Fluid Mechanics*, 1956.

[19] C. Norberg, “Flow around a circular cylinder: aspects of fluctuating lift,” *Journal of Fluids and Structures*, 2001.

[20] M. J. Morrell and J. D. Reiss, “Inherent Doppler properties of spatial audio,” in *Audio Engineering Society Convention 129*, 2010.

[21] N. Jillings *et al.*, “Web audio evaluation tool: A browser-based listening test environment,” *Sound and Music Computing*, 2015.

ELECTRONIC HEARING PROTECTION FOR MUSICIANS

Robert Albrecht
Aalto University
Department of Computer Science
robert.albrecht@aalto.fi

Jussi Jaatinen
University of Helsinki
Musicology
jussi.o.jaatinen@helsinki.fi

Tapio Lokki
Aalto University
Department of Computer Science
tapio.lokki@aalto.fi

ABSTRACT

Many people are exposed to large sound pressure levels either occasionally or regularly, and thus need to protect their hearing in order to prevent hearing loss and other hearing disorders. Earplugs are effective at attenuating sound from the environment, but they do not attenuate bone-conducted sound, but instead amplify it at low frequencies due to the occlusion effect. This is a problem, e.g., for many musicians and especially wind instrument players, since this low-frequency amplification greatly affects the sound of their own instruments. This makes it difficult for the musicians to play while using hearing protectors, and therefore many musicians choose not to use hearing protectors at all. In this paper, we propose electronic hearing protectors that mitigate the problems associated with musicians' hearing protection through several different approaches: reduction of the occlusion effect, adjustable attenuation with natural timbre, and monitoring of the musician's own instrument. We present the design of prototype electronic hearing protectors and the evaluation of these by professional musicians, where they were shown to alleviate the problems associated with conventional hearing protectors.

1. INTRODUCTION

Musicians, among others, are exposed to large sound pressure levels that may over time lead to hearing loss, hyperacusis, and tinnitus. Since the sound pressure caused by the musicians' instruments or the PA system often cannot be limited to safe levels during rehearsals and especially during concerts, personal hearing protection must be used to limit the noise exposure. Hearing protection – typically earplugs when it comes to performing musicians – is effective for attenuating the sound of other musicians playing in an ensemble. However, plugging the ears with hearing protectors makes the sound of the musician's own instrument unnatural. The main reason for this is that part of the sound reaches the inner ear through bone conduction. This bone-conducted sound is not attenuated by the earplugs, but instead amplified by them, and is often very different from the normal air-conducted sound of the instrument.

This is a problem, e.g., for wind instrument players, who get a large degree of bone-conducted sound from their instruments. Double-reed instruments, such as the oboe and the bassoon, are especially problematic, since the vibrating mouthpiece produces a buzzing sound that also disturbs the musician's perception of the pitch of his or her own instrument. Also violinists are affected, since they rest the instrument against their jaws. Singers are equally affected by the problem, as is any musician or other person speaking while wearing hearing protection.

An earplug-occluded ear canal amplifies bone-conducted sound up to 30 dB at low frequencies due to the occlusion effect [1]. Together, the attenuation of air-conducted sound and the amplification of bone-conducted sound causes the sound of the instrument to appear distorted to the musician him- or herself. For this reason, as well as other reasons, many musicians are reluctant to use hearing protection [2] and therefore often suffer from hearing disorders [3]. Another problem when using earplugs is the change in timbre when the ear canal is blocked [4].

In this paper, we present the design and evaluation of an electronic hearing protection solution aimed particularly at musicians. The solution consists of noise-attenuating insert headphones with microphones both inside the ear canal and on the outside of the headphones. The in-ear microphones are used for partial cancellation of the occlusion effect with the help of a negative feedback circuit, while the microphones outside can be used for hear-through purposes, resulting in hearing protectors with adjustable attenuation. Additional microphones are used to pick up the sound of the instrument. This sound can then be reproduced at a desired level, in order to restore some of the natural balance between the air-conducted and the bone-conducted sound of the instrument.

2. BACKGROUND AND RELATED RESEARCH

In a survey among five major classical orchestras in the Helsinki region, Finland, 31% of the 196 musicians that answered the survey reported some degree of hearing loss, 37% reported temporary tinnitus (15% of the women and 18% of the men reported permanent tinnitus), and 43% reported having hyperacusis [2]. These hearing problems were also associated with a high level of stress: the musicians with hearing problems had three to nine times more stress than the other musicians.

In the same survey, 69% of the musicians were at least somewhat worried about their hearing. However, only 6% of the musicians always used hearing protectors during rehearsals and performances. Musicians often only start using hearing protectors once they have developed hearing

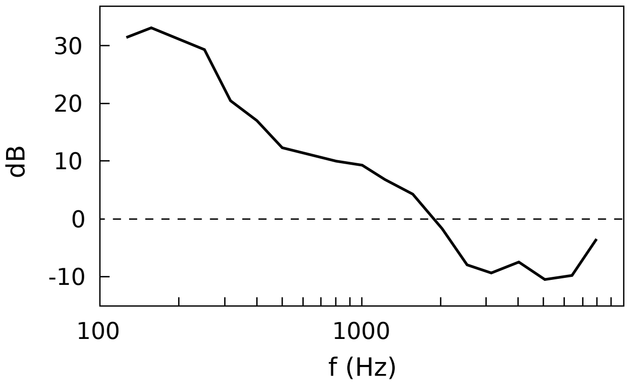


Figure 1: The occlusion effect measured as the change in sound pressure when blocking the ear canal with a shallowly inserted foam earplug [1]. The excitation signal was produced by a bone-conduction transducer.

problems: 20% of the musicians with hearing problems used hearing protectors, while 6% of the musicians without hearing problems used hearing protectors. Among the musicians that answered the questionnaire, 47% used custom-molded earplugs, while 25% used disposable earplugs.

According to the survey, the main reason why many classical musicians do not use hearing protectors is that it hinders their own performance (n=155). The second largest reason is that earplugs make it difficult to hear other musicians in an ensemble (n=88). Further reasons are that the sensation of hearing protectors is unpleasant (n=15) and that they are difficult to insert (n=12).

One specific problem when using hearing protectors both hindering the musician's own performance and making it difficult to hear others play is the occlusion effect. In a normal situation the ear canal is open and the bone-conducted sound energy transmitted from the musician's own instrument into the ear canal through vibration of the ear canal walls exits to a large degree through the ear canal opening. When the ear canal is blocked with an earplug, much of the bone-conducted sound energy is directed to the eardrum. This causes the amplification of low-frequency sound, which is referred to as the occlusion effect. Another way to describe the occlusion effect is that the open ear canal acts as a high-pass filter, but occluding the ear canal removes this high-pass filter which causes low frequencies to become louder [1].

Figure 1 shows the occlusion effect measured by the increase in sound pressure when occluding the ear canal with a shallowly inserted foam earplug. As shown, the sound pressure produced by a bone-conduction transducer is amplified approximately 30 dB at frequencies between 100 and 200 Hz. Above these frequencies, the occlusion effect gradually decreases: at 1 kHz, the amplification is approximately 10 dB, while no amplification is observed at 2 kHz and above.

The low-frequency boost caused by the occlusion effect causes two problems for musicians using earplugs. Firstly, it colours the sound, making it darker and boomy. This makes it difficult for the musicians to hear the nuances of the sound of their own instruments. Secondly, the boost makes the musicians' own instruments sound louder when compared to the sound of the rest of the orchestra. This makes it difficult for the musicians to hear the rest of the orchestra, and to balance the dynamics of their own instruments with the orchestra.

There are several different approaches that can be utilized in order to mitigate the occlusion effect. Inserting earplugs deeper into the ear canals reduces the area of the ear canal walls that radiates bone-conducted sound and thereby also reduces the occlusion effect [1]. Deep earplugs are, however, often perceived as uncomfortable. On the other hand, using large earmuffs instead of earplugs also reduces the occlusion effect [1]. These are usually not used by performing musicians due to aesthetic reasons. Finally, vents can be used to allow the low-frequency energy in the ear canal to escape. This might be a useful approach in hearing aids, but in hearing protectors it will not only reduce the occlusion effect but also reduce the effectiveness of the hearing protection at low frequencies.

A further approach is to utilize noise-cancellation techniques in order to cancel out the bone-conducted sound radiating into the ear canal. In this case, the bone-conducted sound is recorded inside the ear canal and simultaneously played back from the earplug in opposite phase. The resulting destructive interference reduces the sound pressure inside the ear canal and thereby also reduces the occlusion effect.

Several previous studies have tried and succeeded in reducing the occlusion effect caused by an earplug using miniature microphones and loudspeakers inside the ear canal. The majority of this research has been done in the field of hearing aids. Mejia et al. [5] combined analog negative feedback with an acoustic vent, and were able to reduce the occlusion effect in hearing aids by 15 dB. Test subjects reported that their own voice was more natural when using this occlusion reduction.

Borges et al. [6] developed an adaptive occlusion canceller for hearing aids, obtaining an average attenuation of 6.3 dB. The test subjects reported that the occlusion canceller created a sensation of their "ears opening." Sunohara et al. [7] also proposed an adaptive system for reducing the occlusion effect. Computer simulations of the proposed algorithm showed a maximum occlusion reduction of 30 dB. However, due to the adaptive nature of the algorithm, the effectiveness of the reduction varied over time.

Bernier and Voix [8] proposed a hearing protection solution for musicians similar to the one presented in this paper, but without monitoring of the musician's own instrument. Analog negative feedback was used to reduce the occlusion effect, while signals from microphones on the outside of the hearing protectors were processed with a digital signal processor and reproduced through the miniature loudspeakers inside the ear canals in order to obtain adjustable attenuation with natural timbre. The solution was able to achieve a reduction of the occlusion effect of approximately 10 dB. Bernier and Voix also proposed to compensate for the non-linearity of loudness perception in order to make the timbre independent of the current attenuation of the hearing protectors.

3. TECHNICAL SOLUTION

Our electronic hearing protection solution combines several different techniques in order to solve the problems mentioned earlier. The whole solution is illustrated with a block diagram in Fig. 2, while Fig. 3 depicts the location of the microphones connected to the hearing protectors. The designed system consists of the following parts:

- Insert headphones are utilized as earplugs to provide passive attenuation of air-conducted sound.
- Microphones inside the ear canals are used to reduce the occlusion effect through negative feedback.
- Microphones outside the headphones pick up sound from the environment which is reproduced through the headphones at a desired level, resulting in the hearing protectors having adjustable attenuation.
- Additionally, the musician's own instrument can be close-miked to provide in-ear monitoring of the instrument, increasing the ratio of air-conducted to bone-conducted sound and thus providing the musician with a more natural sound of the instrument.

In the current solution, Akustica AKU143 MEMS microphones are used. The signals from the hear-through and instrument microphones are processed at a sampling rate of 48 kHz on an Analog Devices ADAU1442 digital signal processor (DSP). The headphones used are Sennheiser OCX 686G insert headphones.

3.1 Occlusion reduction

As stated earlier, there are many approaches available to alleviate the occlusion effect, but all of them have their disadvantages. Our solution is an analog electroacoustic negative feedback circuit, which partly cancels out the amplification of low frequencies caused by the occlusion effect. The feedback loop inverts the signal and applies a second-order low-pass filter at 1.1 kHz to avoid instability at higher frequencies. The processing is performed with analog electronics in order to minimize the phase shift and allow the occlusion reduction to be effective at as high frequencies as possible.

The chosen approach not only attenuates low frequencies amplified by the occlusion effect, but it attenuates low frequencies inside the ear canal independent of their origin. The negative feedback loop thus also provides improved attenuation of low-frequency air-conducted sound.

3.2 Adjustable attenuation and natural timbre

Employing microphone hear-through techniques [4,9] provides the possibility to adjust the amount of attenuation, from the full attenuation provided by the insert headphones tightly fit in the ear canals, together with the occlusion reduction, to no attenuation at all. The amount of attenuation can thus be chosen for different situations according to the need of the user.

The microphone hear-through technique utilizes microphones attached to the outside of the headphones, one at each ear. The signals from these microphones can be amplified to the desired level, and reproduced through the headphones. The microphones should be placed as close as possible to the ear canal entrances, to avoid colouration of the sound picked up by them. To achieve natural timbre, the signals must also be equalized, in order to compensate for the magnitude response of the headphones as well as the changes in resonances when blocking the ear canals. For this task, the DSP was used. The headphone response with the occlusion reduction circuit active was first flattened using four notch filters and one peak filter, to produce an approximately flat response at the eardrum inside an ear canal simulator. The quarter-wavelength resonance present in an open ear canal was then added using a peak filter, together with the three-quarter-wavelength resonance using another peak filter. The filters were tuned to match the target response proposed by Hoffmann et al. [10].

3.3 Monitoring of the musician's own instrument

The occlusion reduction circuit eliminates much of the amplification of low-frequency bone-conducted sound caused by occluding the ear canals. However, even if all of this amplification were eliminated, the attenuation of the air-conducted sound would cause the bone-conducted sound to be more prominent than with unoccluded ear canals. The timbre of the musician's own instrument would thus still sound unnatural. To compensate for this imbalance, we use up to four microphones attached to the instrument. These microphones pick up the air-conducted sound of the instrument that can be appropriately amplified and reproduced through the headphones to achieve a satisfactory balance between bone-conducted and air-conducted sound and thus a more natural timbre. The maximum of four instrument microphones was at this stage chosen to give enough possibilities to try different microphone setups with different instruments.

It is, however, not adequate to simply reproduce these microphone signals as such. First of all, the air-conducted sound heard from the instrument is normally affected by the body and especially the head and pinnae of the musician, altering the spectral and temporal characteristics of the sound that enters each ear. These modifications to the sound serve as cues for the human auditory system to infer that the sound source is in a specific direction. If these modifications are absent, the sound will be perceived to originate from inside the head, and it will also have an unnatural timbre.

Second, sound from the instrument normally also reaches the musician's ears through reflections from different surfaces in the surrounding space. Since the microphones are placed close to the instrument, in order to pick up the sound of this instrument and as little as possible of the other musicians' instruments, they will also pick up very little of the reflections from the environment. Without these naturally occuring reflections, the sound of the instrument will feel "dry" and "out of place," and the absence of reflections will also make it more likely that the sound of the instrument is

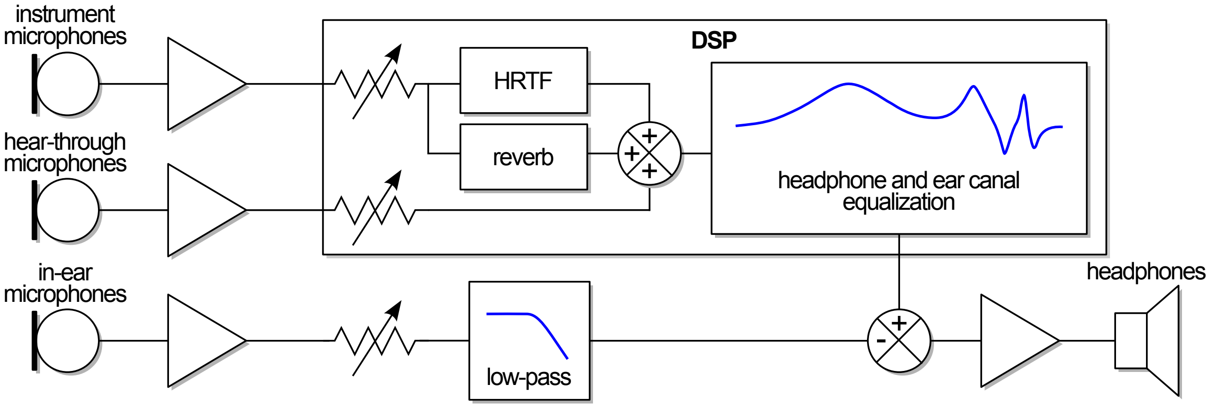


Figure 2: Block diagram of the electronic hearing protection solution.

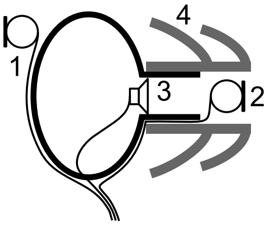


Figure 3: Diagram of the hearing protectors and the attached microphones: 1. hear-through microphone, 2. in-ear microphone. 3. headphone driver, and 4. rubber tip.

perceived as originating inside the musician's head.

To solve the mentioned problems, two different approaches were combined. First, head-related transfer functions (HRTFs) were applied to the mixed instrument microphone signals. HRTFs describe the effects that the head and torso have on the sound reaching each ear from a certain direction, under otherwise anechoic conditions. With HRTFs applied to the sound of the instrument, it will sound as if arriving from a certain direction, due to the HRTFs containing different cues that the auditory system uses for sound source localization. Additionally, since people normally perceive sound through HRTFs, so to speak, they are accustomed to the timbre that these produce, so applying HRTFs will make the timbre more natural.

In addition to using HRTFs, we employ reverberation in order to integrate the "dry" sound of the instrument into the surrounding acoustic environment. For the evaluation, we utilized the reverberation algorithm readily available on the DSP, and adjusted the few available parameters to fit it to the surrounding environment as well as possible. Adding reverberation not only helps with integration, but also aids in externalization [11].

3.4 Details on HRTFs

Because people have different sizes of heads and ears, ideally the individual HRTFs of the person in question should be used, in order to achieve totally natural localization cues and timbre. Measuring individual HRTFs is, however, currently not feasible on a large scale, so generic HRTFs were instead chosen here. For the evaluation, we used near-field HRTFs of a KEMAR head and torso simulator [12]. To represent the direction of many wind instruments, with respect to the head of the musician, an elevation of -30° was chosen. An azimuth of -10° was chosen in favour of an azimuth of 0° (straight forward), since HRTFs measured at 0° azimuth often reduce externalization [11], i.e., sounds presented with them are perceived as originating from inside the head. The HRTFs were measured at a distance of 20 cm from the centre of the head, thus simulating a sound source a small distance in front of the mouth.

Naturally, different instruments are played in different positions, so the HRTFs should be selected based on the instrument. If the instrument is not always kept in the same direction with respect to the head, tracking of the instrument position might be beneficial. With most wind instruments, however, this should not be a problem, since they are normally kept in the same direction with respect to the head. Although the chosen azimuth and elevation of the HRTFs may not correspond perfectly with the location of the instrument, this should not be a problem unless the mismatch is large. Due to the ventriloquism effect [13], small mismatches are ignored and the sound should be heard as if coming from the instrument.

4. VALIDATION

The prototype electronic hearing protection system constructed in order to evaluate the proposed solution is shown in Fig. 4. Measurements were performed in order to quantify the effect of the occlusion reduction circuit, as well as for tuning the headphone equalization. To evaluate the sound of the electronic hearing protectors, they were tested by seven professional musicians.

4.1 Measurements

For the measurements, we constructed an ear canal simulator out of a silicone tube with an inner diameter of 10 mm and a length of 27 mm. One end of the tube was glued to a piece of hard plastic, simulating the eardrum, and a MEMS microphone was attached to the plastic. The magnitude response of the headphones under different conditions was measured using white noise.

Figure 5 shows the magnitude response of the headphones without and with the occlusion reduction circuit

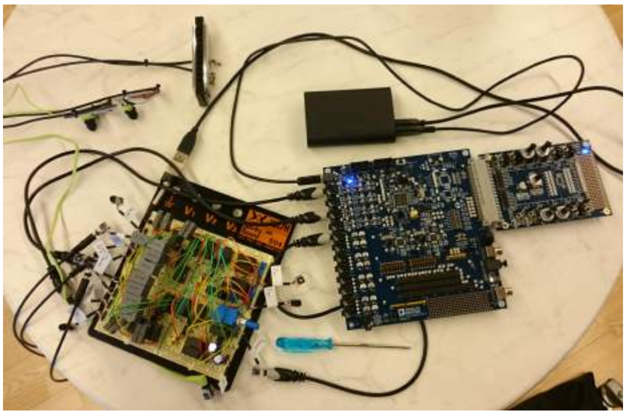


Figure 4: The prototype electronic hearing protection solution, consisting of a breadboard with analog electronics, a DSP evaluation board, a USB battery pack, headphones with attached microphones, and a harmonica with two attached instrument microphones.

in action. Figure 6 shows the reduction of the occlusion effect achieved with the occlusion reduction circuit. The occlusion reduction is at most approximately 13 dB at 150 Hz, and more than 10 dB between 100 and 300 Hz. At the same time the peak in the headphone response between 5 and 6 kHz is amplified since the feedback loop is not in opposite phase any more at this frequency. Much further amplification in the feedback loop results in an even larger peak and finally instability at this frequency. The amplification of the feedback loop was thus chosen as a compromise between attenuation at low frequencies and amplification and stability at high frequencies.

Frequencies below 50 Hz are amplified, with a maximum amplification of 12 dB. This low-frequency boost is due to the phase shift introduced by DC-blocking capacitors in the occlusion reduction feedback loop. Future design efforts should try to mitigate this problem, e.g., by using larger capacitances that would shift these high-pass filters to lower frequencies and reduce the boost.

Figure 7 depicts the equalization of the headphone response, which is applied to the signals from both the hear-through microphones and the instrument microphones (as shown in Fig. 2). The equalized response resembles the target response proposed by Hoffman et al. [10], and should thus provide a fairly natural timbre for the signals reproduced through the headphones. Note that since the occlusion reduction circuit affects not only the bone-conducted sound entering the ear canal, but also the signals that are reproduced through the headphones, the headphone equalization must compensate for the notch caused by the occlusion reduction.

4.2 Subjective evaluation

The prototype electronic hearing protection solution was informally evaluated by seven professional musicians. One of the participants was a jazz musician, while the rest were members of either the Helsinki Philharmonic Orchestra or the Finnish National Opera. All musicians tested the hearing protectors with their own instruments, which were the tenor saxophone, viola, bassoon, trumpet, flute and piccolo, clarinet, and oboe. The evaluation was performed in the listening room of the Department of Computer Science, where different orchestral compositions could be reproduced as if performed in different concert halls (see, e.g., [14]). This gave the musicians the possibility to play and evaluate the hearing protectors with a virtual orchestra.

The goal of the evaluation at this stage was not to accurately prove the effectiveness of the proposed solution, but rather to make sure that the basic concept works, and to identify the future directions for developing the solution further. The musicians were thus allowed to freely play without and with the hearing protectors, with the different features (occlusion reduction, hear-through, monitoring) enabled or disabled, and with or without the orchestra recordings. The evaluation results were documented based on a free discussion between the authors and the musicians.

4.2.1 The need for and use of hearing protectors

Some of the musicians reported music-induced hearing loss and tinnitus, while others did not have any hearing disorders. All musicians recognized the need for hearing protection in their work, some occasionally while others more often. A few musicians mentioned that their need for hearing protection had decreased due to, e.g., seating arrangements in the orchestra.

When using hearing protection, the musicians typically used foam earplugs, and some often only in one ear. The trumpet player said that he felt a strong need for using hearing protectors, but that he never uses them when playing in an orchestra, since the booming sound caused by the earplugs overpowers the sound of both the rest of the orchestra and his own instrument. The other musicians mentioned different reasons for why they do not use hearing protectors more often: the sound of the instrument is bad; trebles get cut off; buzzing and other unwanted sounds get amplified and annoying; it takes a lot of time and effort to get used to playing with earplugs; you feel isolated from the rest of the orchestra; it is difficult to play in balance with the rest of the orchestra.

4.2.2 Occlusion reduction

The occlusion reduction had a different effect depending on the instrument. The trumpet player said that the booming sound caused by the earplugs vanished, and that he now was able to hear himself playing. The saxophonist reported a small but clear improvement of the instrument sound. The bassoonist said that the sound in some cases – depending on the note played – was more natural, but that it otherwise just made the sound different and more distant. The viola player felt that the timbre of the instrument became too bright. The clarinetist reported a clear change in the timbre, but it did not improve the sound; the unwanted sounds that were amplified by the earplugs remained. The flutist said that the sound became brighter and more spacious, which was an improvement over the stuffy sound caused by the earplugs.

The occlusion reduction comes with a noticeable degree of noise. During the evaluation, however, this noise was

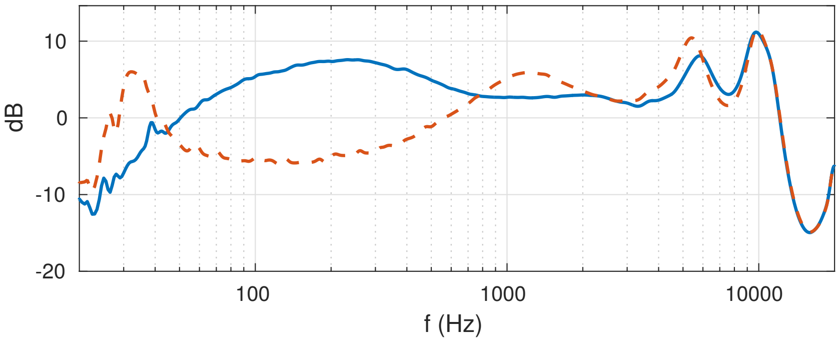


Figure 5: Magnitude response of the headphones, measured in the ear canal simulator with the eardrum microphone. The solid blue line represents the unprocessed response, while the dashed red line represents the response with the occlusion reduction circuit active.

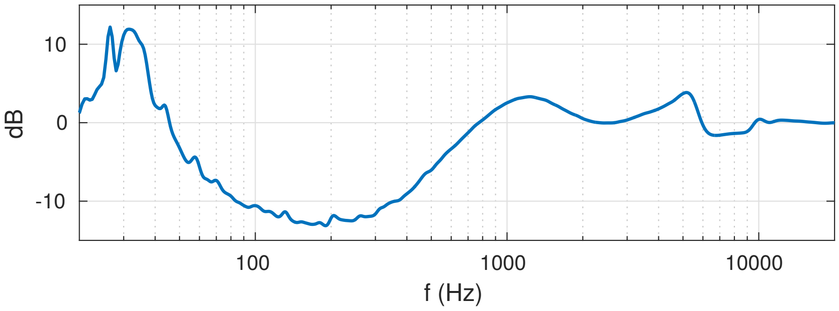


Figure 6: The effect of the occlusion reduction circuit on the magnitude response of the headphones, i.e., the difference between the solid blue line and the dashed red line in Fig. 5.

considered not to be loud enough to be disturbing, and the spectrum of the noise was also considered not to be disturbing.

4.2.3 Monitoring

The placement of the instrument microphones was chosen based on experimentation to provide a balanced and natural sound of the instrument. Although the prototype hearing protection solution supports connecting up to four instrument microphones, only two microphones were found to be necessary during the evaluation. These two microphones were in the end placed closed to each other, since this provided a balanced sound of all the instruments tested. On the viola, the microphones were placed at the bridge. On the bassoon, trumpet, saxophone, oboe, and clarinet, the microphones were placed at the side of the bell. On the flute and piccolo, the microphones were placed approximately one-third of the instrument's length from the end of the instrument closest to the mouthpiece.

Amplifying the sound of the instrument microphones, processed with HRTFs and with reverberation added, clearly improved the sound of the instrument. Most musicians commented that the sound was not completely natural, but quite pleasant. In many cases, slightly amplifying the equalized signal from the hear-through microphones further improved the sound.

Some of the musicians thought that the reverberation added to the sound was pleasant, while others felt that it was unnatural, or that there was either too little or too much reverberation. Clearly, the level and quality of reverberation must be easily adjustable depending on the surrounding space, the instrument, and the preferences of the musician.

4.2.4 Playing alone

Especially the trumpet player, the saxophonist, and the piccolo player thought that the electronic hearing protectors would be good for practicing alone. When practicing in a small room, the sound of the instrument is often too loud due to strong reflections. With the electronic hearing protectors, however, the sound of the instrument can be attenuated, while still remaining quite clear and pleasant, unlike when using normal earplugs. The piccolo player summarized the experience: “It sounds good and natural, but it doesn't hurt, like it usually does.”

4.2.5 Playing with an orchestra

The musicians tried the electronic hearing protectors while listening to a symphony orchestra recording and playing their instruments. The musicians commented that they were able to hear both their own instrument and the rest of the orchestra well. The flutist also commented that the sound of her own instrument was richer and more inspirational when playing with the electronic hearing protectors compared with regular earplugs. The saxophonist said that he could hear his instrument well and play with the orchestra, but wondered how he would be able to adjust his dynamics with respect to the rest of the orchestra, since

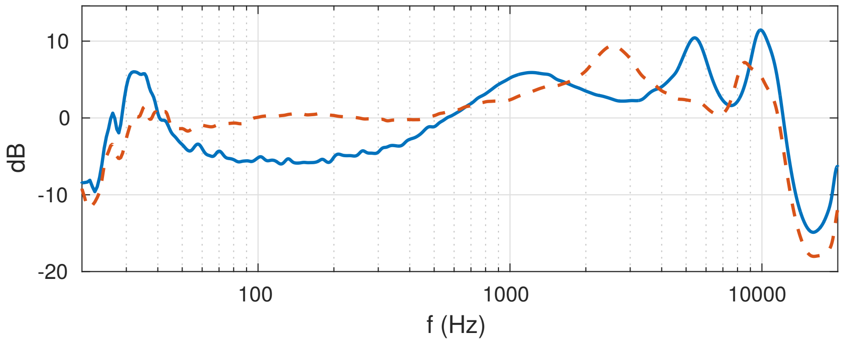


Figure 7: Equalization of the headphone response when reproducing the signals from the hear-through and instrument microphones. The solid blue line represents the unequalized magnitude response with the occlusion reduction circuit active. The dashed red line represents the equalized response. The measurements were performed in the ear canal simulator with the eardrum microphone.

the electronic hearing protectors alter the level of the own instrument and the orchestra independently.

4.2.6 Technical implementation

The rubber tips of the headphones did not fit well in the ears of all the musicians, and thus different sized tips would naturally be needed for different ear canal sizes. The need for custom ear molds also came up.

Based on the musicians' comments, the sound of the hearing protectors (timbre, balance between the musician's own instrument and the orchestra, and reverberation) should be easily adjustable, but preferably minimal adjustments should be needed to get the sound right. A few musicians also pointed out that it might be beneficial to adjust the hear-through level separately for each ear, since sometimes there are loud instruments that should be attenuated only on one side. Naturally, this would affect localization cues and the spatial perception of the auditory scene, and experiments should be performed to deterimine if such an effect is acceptable.

Based on the evaluation, the instrument microphones could be mounted on so called goosenecks, allowing easy adjustment of the microphone position. A single microphone might be enough in many cases, and seemed to provide a pleasant sound with the instruments in this evaluation (we actually used two microphones in all cases, but they were positioned very close to each other).

4.3 Discussion

The subjective evaluation of the prototype hearing protection system showed that the occlusion reduction in many cases improves the experience for the musician, by attenuating amplified bass frequencies and thus providing a timbre that is more pleasant and natural. With the ears plugged, the sound of the instrument can often appear as if it originates from inside the head, but in some cases the occlusion reduction can alleviate this problem.

The source of the noise that is heard when the occlusion reduction circuit is enabled is currently unclear. Unless the source is the self noise of the microphones, it could probably be reduced with more advanced circuit design. Another problem with the current occlusion reduction circuit design is the low-frequency boost that can be seen in Fig. 6. Although few musical instruments have energy at such low frequencies, this boost can, e.g., emphasize sound caused by the movement of the hearing protector cables, and should thus be addressed in future designs, if possible.

Adding the instrument microphone signals with HRTFs and reverberation makes the timbre brighter and more natural, and also makes the sound of the instrument seem as it originates from the instrument and not from inside the head. For most wind instruments, a bone-conducted buzzing sound from the reed or lips is normally prominent when playing with the ears occluded, making playing with ear plugs uncomfortable for many musicians. Adding the instrument microphone signals alleviates this problem, by altering the balance between air-conducted and bone-conducted sound to be more favourable and closer to natural.

Amplifying the instrument microphone signals will, however, affect the balance between the musician's own instrument and the other musicians' instruments when playing in an ensemble. The musician's instrument will thus sound louder than normal when compared with the sounds of other instruments. This balance will be, among other things, instrument specific, since all instruments have a different natural balance between air-conducted and bone-conducted sound. The balance between the musician's own instrument and the other instruments can of course be improved by amplifying the signals from the hear-through microphones. However, heavy amplification of these signals will counteract the function of the hearing protectors.

Adding reverberation to the instrument signals may be useful not only to aid in externalization and integrating the sound of the instrument into the surrounding acoustic environment. It can also be used for personal practice, making the instrument sound like it's being played in, e.g., a much larger hall. The possibilities to adjust the reverberation algorithm available on the DSP are, however, limited, and the available memory as well as the graphical development tool for programming the DSP limit the possibilites to develop other algorithms. Thus, a more versatile DSP would be needed in the future. Even with more sophisticated reverberation algorithms, the task of combining real and artificial reverberation is far from trivial. To minimize the amount of manual adjustment needed, impulse sounds could be isolated from the hear-through microphone signals, and the characteristics of the real reverberation extracted from these.

The delay introduced by the DSP in the hear-through and instrument microphone signals can cause two different problems. A short delay will result in a comb-filtering effect, as the microphone signals reproduced through the headphones get summed with the sound leaking past the earplugs into the ear canals. The delay itself will become noticeable as it grows larger, affecting the performance of the musician. The allowable delay depends greatly on, e.g., the instrument type, but delays greater than 1 ms may already produce audible comb-filtering effects and delays greater than 6.5 ms can be perceived as audible delays with some instruments [15]. The situation gets more complicated as the sound of the other musicians' instruments also are delayed, and especially if several musicians are using similar hearing protectors. The delays of the prototype hearing protection solution were not measured. However, none of the musicians reported that the delay negatively affected their performance.

5. CONCLUSIONS

In this paper, we presented an electronic hearing protection solution designed especially for musicians. The solution combines the following features: reduction of the occlusion effect, monitoring of the musician's own instrument, adjustable attenuation, and natural timbre. Seven professional musicians evaluated the implemented solution and confirmed that these features together alleviate problems associated with musicians' hearing protection. Thus, the findings presented in this paper will hopefully lead to better hearing protectors in the future and more musicians that are able to satisfactorily protect their hearing.

Acknowledgments

The work leading to these results was partially funded by the Finnish Work Environment Fund, project no. 115145. The authors would like to thank the musicians that participated in the evaluation of the proposed hearing protection solution. The authors would also like to thank professor emeritus Jukka Ylikoski for consultation at the beginning of the project.

6. REFERENCES

[1] S. Stenfelt and S. Reinfeldt, "A model of the occlusion effect with bone-conducted stimulation," *International Journal of Audiology*, vol. 46, no. 10, pp. 595–608, 2007.

[2] H. Laitinen, "Factors affecting the use of hearing protectors among classical music players," *Noise & Health*, vol. 7, no. 26, pp. 21–29, 2005.

[3] K. R. Kähäri, "The influence of music on hearing. a study in classical and rock/jazz musicians," Ph.D. dissertation, University of Gothenburg, Sweden, 2002.

[4] R. Albrecht, T. Lokki, and L. Savioja, "A mobile augmented reality audio system with binaural microphones," in *Proceedings of the Interacting with Sound Workshop*, Stockholm, Sweden, 2011, pp. 7–11.

[5] J. Mejia, H. Dillon, and M. Fisher, "Active cancellation of occlusion: An electronic vent for hearing aids and hearing protectors," *The Journal of the Acoustical Society of America*, vol. 124, no. 1, pp. 235–240, 2008.

[6] R. Borges, M. Costa, J. Cordioli, and L. Assuiti, "An adaptive occlusion canceller for hearing aids," in *Proceedings of the 21st European Signal Processing Conference*, Marrakech, Morocco, 2013.

[7] M. Sunohara, K. Watanuki, and M. Tateno, "Occlusion reduction system for hearing aids using active noise control technique," *Acoustical Science and Technology*, vol. 35, no. 6, pp. 318–320, 2014.

[8] A. Bernier and J. Voix, "Active musician's hearing protection device for enhanced perceptual comfort," in *EuroNoise 2015*, Maastricht, the Netherlands, 2015, pp. 1773–1778.

[9] R. W. Lindeman, H. Noma, and P. G. de Barros, "Hear-through and mic-through augmented reality: Using bone conduction to display spatialized audio," in *Proceedings of the 6th International Symposium on Mixed and Augmented Reality*, Nara, Japan, 2007, pp. 173–176.

[10] P. F. Hoffmann, F. Christensen, and D. Hammershøi, "Insert earphone calibration for hear-through options," in *51st International Audio Engineering Society Conference*, Helsinki, Finland, 2013.

[11] J. Catic, S. Santurette, and T. Dau, "The role of reverberation-related binaural cues in the externalization of speech," *The Journal of the Acoustical Society of America*, vol. 138, no. 2, pp. 1154–1167, 2015.

[12] T. Qu, Z. Xiao, M. Gong, Y. Huang, X. Li, and X. Wu, "Distance-dependent head-related transfer functions measured with high spatial resolution using a spark gap," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 6, pp. 1124–1132, 2009.

[13] W. R. Thurlow and C. E. Jack, "Certain determinants of the 'ventriloquism effect'," *Perceptual and Motor Skills*, vol. 36, no. 3 suppl., pp. 1171–1184, 1973.

[14] T. Lokki, J. Pätynen, A. Kuusinen, and S. Tervo, "Concert hall acoustics: Repertoire, listening position, and individual taste of the listeners influence the qualitative attributes and preferences," *The Journal of the Acoustical Society of America*, vol. 140, no. 1, pp. 551–562, 2016.

[15] M. Lester and J. Boley, "The effects of latency on live sound monitoring," in *Audio Engineering Society Convention 123*, New York, NY, USA, 2007, paper no. 7198.

VIRTUAL ANALOG SIMULATION AND EXTENSIONS OF PLATE REVERBERATION

Silvin Willemsen
Aalborg University
swille15@student.aau.dk

Stefania Serafin
Dept. AD:MT
Aalborg University
sts@create.aau.dk

Jesper Rindom Jensen
Audio Analysis Lab, AD:MT
Aalborg University
jrj@create.aau.dk

ABSTRACT

In the 50s and 60s, steel plates were popularly used as a technique to add reverberation to sound. As a plate reverb itself is quite bulky and requires lots of maintenance, a digital implementation would be desirable. Currently, the available (digital) plugins rely solely on recorded impulse responses or simple delay networks. Virtual Analog (VA) simulations, on the other hand, rely on a model of the analog effect they are simulating, resulting in a sound and 'feel' the of the classical analog effect. In this paper, a VA simulation of plate reverberation is presented. Not only does this approach result in a very natural sounding reverb, it also poses many interesting opportunities that go beyond what is physically possible. Existing VA solutions, however, have limited control over dynamics of physical parameters. In this paper, we present a model where parameters like the positions of the in- and outputs and the dimensions of the plate can be changed while sound goes through. This results is in a unique flanging and pitch bend effect, respectively, which has not yet be achieved by the current state of the art.

1. INTRODUCTION

A great number of digital audio effects is currently available to musicians and producers. Many sounds we could not even imagine a few years ago, we can now create using current DSP technology. However, despite this immense amount of options, there is still a great desire for the sound of the classical analog effects that made their first appearance in the late 40s and characterised music from the 50s and 60s onwards. Even though, generally, digital sound effects 'do the job', they do not have the 'feel' that the old analog effects had - something greatly desired by many musicians. Contrary to digital effect simulations, Virtual Analog (VA) simulations rely on a model of the analog effect they are simulating [1]. A great advantage of VA simulations over the original systems is that they do not age and thus do not require time consuming maintenance. Also, when digitalised they are easily accessible, mostly simpler to use and can be made much cheaper than their analog counterparts. Naturally, the sound of a used analog audio effect can have its charm, but if desired, this can be modelled into the simulation. Also, VA simulations make it possible for parameters like room size, material properties, etc., to be changed, which is physically impossible or very hard to do. This can result in unique sounds that can only be created using VA simulations.

A popular reverberation technique used in the 50s and 60s was plate reverberation. A plate reverb utilises a small speaker (actuator) attached to a big steel plate to make it vibrate, and several pickups to pick up the sound after it has propagated through the plate. Several different plate reverbs made it to the market, the most popular being the EMT140 used in the Abbey Road Studios and undoubtedly leaving a mark on music in the aforementioned years. In fact, it was the only reverb used on Pink Floyd's Dark Side of the Moon [2]. A big issue of using an actual plate reverb is the sheer size and weight of it. The plate is 2x1m big and weighs (together with the rest of the installation) roughly 270kg [3], hence, a digital implementation of it would be desirable.

There are different ways of digitally implementing plate reverberation. Convolution is an effective approach, but not a VA one as it is not based on physical parameters. This means that flexibility is limited. Feedback Delay Networks (FDNs), as proposed by Jean-Marc Jot in [4], are also used as an implementation approach for creating a digital plate reverb by Jonathan Abel in [5]. FDNs are an efficient way of realising a VA solution to the plate reverberation problem. However, in his implementation, Abel uses a hybrid structure consisting of a convolutional part and an FDN-based part, making it not fully VA. Finite difference schemes as proposed in [6] and [7] are flexible, VA, but very computationally heavy solutions. Lastly, the vibrations of the plate can be decomposed into a series of plate modes - a modal description of the plate. It is both a VA and not computationally heavy approach and gives a lot of freedom in dynamic parameter manipulation.

Currently there are some VA plate reverb plugins available such as the *PA1 Dynamic Plate Reverb* [8] (that uses the aforementioned modal description) and the *Valhalla-Plate* [9]. These plugins, however, do not use the full potential of VA simulations. Parameters like for example pickup positions and sheet-size can be made dynamic, i.e., changed while sound is going through the plate. This will result in some interesting sounds and effects that can not be achieved with a physical plate reverb. In this paper we propose a VA simulation of plate reverberation that utilises these dynamic parameters - something that the aforementioned plugins have not implemented. Furthermore, different kinds of damping that occur in physical plate reverbs are taken into account to make the simulation sound even more natural.

This work is structured as follows: in Section 2 the physics of a thin metal plate will be explained, in Section 3 a numerical solution will be described, in Section 4 the results of this solution will be discussed and lastly in Section 5 we will conclude and discuss future works for this project.

2. PHYSICS OF A THIN METAL PLATE

In order to simulate a plate reverb, a model of the physics of the plate is needed. The Kirchhoff-Love model mathematically describes stresses and deformations in thin plates subjected to external forces. The partial differential equation for an isotropic plate (including damping) is [10]:

$$\frac{\partial^2 u}{\partial t^2} = -\kappa^2 \nabla^4 u - c\frac{\partial u}{\partial t} + f(x_{\text{in}}, y_{\text{in}}, t). \tag{1}$$

Here, $u = u(x, y, t)$, a state variable that describes the transverse plate deflection and is defined for $x \in [0, L_x]$, $y \in [0, L_y]$ and $t \geq 0$, c is a loss parameter, ∇^4 is the biharmonic operator, $f(x_{\text{in}}, y_{\text{in}}, t)$ is the input signal at input source location $(x_{\text{in}}, y_{\text{in}})$ at time instance t and finally, κ^2 can be referred to as the stiffness parameter:

$$\kappa^2 = \frac{Eh^2}{12\rho(1 - v^2)}, \tag{2}$$

where E, h, ρ, and v are the Young's modulus, plate thickness, plate density and Poisson's ratio, respectively.

2.1 Frequency dispersion

One feature that characterises the sound of a plate reverb is the fact that higher frequencies propagate faster through a metal plate than lower frequencies [10, 11]. This phenomenon is called frequency dispersion. The dispersion relation is defined as:

$$\omega^2 \simeq \gamma^4 \kappa^2, \tag{3}$$

where ω is the angular frequency and γ the wavenumber. The group and phase velocities are defined as:

$$c_{\text{ph}} = \sqrt{\kappa\omega} \qquad c_{\text{gr}} = 2\sqrt{\kappa\omega}. \tag{4}$$

Using the properties of the EMT140 (steel, 0.5mm thick) and a frequency range of 20-20,000kHz, group velocities can vary between ca. 20-620m/s. In regular (or room) reverberation all frequencies travel at the same speed making frequency dispersion one of the main differences between plate and room reverberation.

2.2 Damping

In plate reverbs, three different kinds of damping occur: thermoelastic damping, radiation damping and damping induced by a porous medium [10, 11]. All three will be shortly described in this section.

2.2.1 Thermoelastic damping

Thermoelastic damping occurs in materials with a high thermal conductivity. It is an internal damping mechanism that damps different frequencies at different strengths according to the following equation [11]:

$$\alpha_{\text{th}}(\omega) = \frac{\omega}{2}\eta(\omega) \approx \frac{\omega^2 R_1 C_1}{2(\omega^2 h^2 + C_1^2/h^2)} \tag{5}$$

where R_1 and C_1 are material dependent constants. In the case of the EMT140 plate reverb, these are $R_1 = 4.94 \cdot 10^{-3}$ and $C_1 = 2.98 \cdot 10^{-4}$ [10].

2.2.2 Radiation damping

Radiation damping happens when vibration is converted to acoustic energy. This happens according to the following equation [10, 11]:

$$\alpha_{\text{rad}} = \frac{1}{4\pi^2}\frac{c_{\text{a}}\rho_{\text{a}}}{\rho h}\frac{2(L_x + L_y)}{L_x L_y}\frac{c_{\text{a}}}{f_{\text{c}}}g(\psi), \tag{6}$$

$$g(\psi) = \frac{(1 - \psi^2)\ln[(1 + \psi)/(1 - \psi)] + 2\psi}{(1 - \psi^2)^{3/2}},$$

where ρ_{a} and c_{a} are the density of air and speed of sound in air respectively and $\psi = \sqrt{\frac{f}{f_{\text{c}}}}$. Here, f_{c} is the critical frequency and can be calculated using $f_{\text{c}} = \frac{c_{\text{a}}^2}{2\pi\kappa}$.

2.2.3 Damping induced by porous medium

Plate reverberators contain a porous plate positioned behind the metal plate. The distance between the two entities can be set creating a difference in low-frequency decay [10, 11]. The damping plate position changes this decay between roughly 0.6 - 5.5 seconds [5] and can be manipulated on the interface of a real plate reverb like the EMT140.

2.3 Boundary conditions

The states of the edges of a plate are referred to as the boundary conditions. In [12], the authors state that a plate can have three different boundary conditions: free, clamped or simply supported (hinged). For a rectangular plate such as the plate reverb this means that there are 27 different possible combinations of boundary conditions. Every combination will have a unique impulse response and will thus sound different. In this work, we limit ourselves to all sides being simply supported. This means that state variable $u = 0$ at $x = 0$, $x = L_x$, $y = 0$ and $y = L_y$.

3. NUMERICAL SOLUTION

The implementation approach we chose to use is the modal description (see Section 1). In this section a numerical solution of this will be presented.

The state u, as seen in the Kirchhoff-Love equation (1) can be modelled as being a summation of a number of different modes [13]:

$$u = \sum_{m=1}^{M}\sum_{n=1}^{N} q_{mn}\Phi_{mn}(x, y), \tag{7}$$

where q_{mn} is the unknown amplitude of mode (m, n) (m being the mode over the horizontal axis and n over the vertical axis of the plate) and Φ_{mn} is a modal shape defined over $x \in [0, L_x]$ and $y \in [0, L_y]$. In this model, $M \cdot N$ is the total number of modes accounted for. In theory, this is infinite. Note that – apart from the chosen time step $(1/f_s)$ – the computational speed depends on M and N.

For a rectangular plate with sides L_x and L_y, using simply supported boundary conditions, $\Phi_{mn}(x, y)$ can be calculated [13]:

$$\Phi_{mn}(x,y) = \frac{4}{L_x L_y} \sin\frac{m\pi x}{L_x} \sin\frac{n\pi y}{L_y}, (m,n) \in \mathbf{Z}^+, \tag{8}$$

and q_{mn} can be found using the following update equation [13]:

$$Aq_{mn}^{t+1} = Bq_{mn}^{t} + Cq_{mn}^{t-1} + \frac{\Phi_{mn}(x_p, y_p)}{\rho h} P^t, \tag{9}$$

where constants A, B and C can be described in the following way:

$$A = \frac{1}{k^2} + \frac{c_{mn}}{\rho h k}, \qquad B = \frac{2}{k^2} - \omega_{mn}^2,$$

$$C = \frac{c_{mn}}{\rho h k} - \frac{1}{k^2}.$$

Moreover, P^t is the input signal at time instance t at specified input location (x_p, y_p), k is the chosen timestep $1/f_\mathrm{s}$ and c_{mn} is a loss coefficient that can be set per eigenfrequency which gives a lot of control over the frequency content of the output sound.

3.1 Eigenfrequencies

The (angular) eigenfrequencies ω_{mn} can be calculated using the following equation [14]:

$$\omega_{mn} = \kappa\pi^2 \left(\frac{m^2}{L_x^2} + \frac{n^2}{L_y^2}\right). \tag{10}$$

According to [13] and experimental observation, stability of the update equation (9) can only be assured if and only if:

$$\omega_{mn} < 2f_\mathrm{s}. \tag{11}$$

This poses a limit on the total number of modes in (7) and a creates the dependency:

$$\omega_{M(n)n}, \omega_{mN(m)} < 2f_\mathrm{s}, \tag{12}$$

where $m \in [1, M(n)]$ and $n \in [1, N(m)]$. All the eigenfrequencies that satisfy this condition will be used in the algorithm.

3.2 Loss coefficients

In [13], the authors set loss coefficients per frequency band. In our implementation, the different damping mechanisms described in Section 2.2 have been implemented to ultimately create a loss coefficient for each individual eigenfrequency. If we set the porous medium to the furthest distance possible, the damping induced by this can be ignored. In Figure 1 the results can be seen.

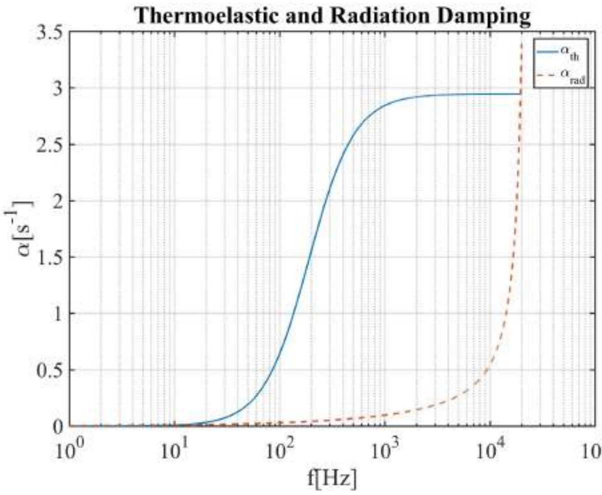


Figure 1. Thermoelastic (α_th) and Radiation damping (α_rad) for values based on the EMT140.

The loss coefficients are then calculated in the following way [11, 13]:

$$c_{mn} = \frac{12\ln(10)}{T_{60}}, \quad \text{where} \quad T_{60} = \frac{3\ln(10)}{\alpha_\mathrm{tot}}, \tag{13}$$

$$\Rightarrow \quad c_{mn} = 4\alpha_\mathrm{tot}.$$

Here, α_tot is simply the addition of all of damping factors.

3.3 Dynamic outputs

The output can be retrieved at a specified point by, in a certain sense, inverting (7):

$$u_\mathrm{out} = \sum_{m=1}^{M} \sum_{n=1}^{N} q_{mn} \Phi_{mn}(x_\mathrm{out}, y_\mathrm{out}). \tag{14}$$

The *PA1 Dynamic Plate Reverb* (as described in Section 1) has the functionality of moving the input from left to right over the plate to create a flanging effect. In our implementation, not only both outputs (not inputs) are able to move, but they are able to move in elliptical and Lissajous patterns. This happens according to the following equations:

$$x_\mathrm{out}(t) = R_x L_x \sin\left(S_x \cdot \frac{2\pi t}{f_\mathrm{s}}\right) + 0.5, \tag{15}$$

$$y_\mathrm{out}(t) = R_y L_y \sin\left(S_y \cdot \frac{2\pi t}{f_\mathrm{s}} + \theta\right) + 0.5. \tag{16}$$

Here, $R_x, R_y \in [-0.5, 0.5]$ when multiplied with L_x and L_y respectively, determine the horizontal and vertical maxima of the output pattern. If one of these has a value of zero, the outputs will move in a linear shape. The shape of the pattern is determined by the horizontal and vertical speeds S_x, S_y and the phase shift θ. For example if

$S_x = S_y = 1$ and $\theta = 0.5\pi$ the outputs will follow an elliptical pattern. The patterns created for different values of S_x, S_y and θ can be seen in [15]. If either S_x or S_y has a value of zero, the outputs will again move in a linear fashion. Note, that it is perfectly possible to set different values for the aforementioned parameters for the left output channel and the right.

3.4 Changing plate dimensions

The fact that VA simulations are extremely flexible, poses a lot of interesting opportunities for model manipulation. We were interested in manipulating parameters that are physically 'fixed', such as the dimensions of a steel plate. Generally speaking, changing the dimensions of a steel plate is physically impossible to do. The challenge was thus to imagine how the sound would change if it would be possible. Looking at the variables that are dependent on the horizontal (L_x) and vertical (L_y) plate dimensions, we see that changing these would change the eigenfrequencies, the modal shapes and radiation damping. As can be seen in (10) the eigenfrequencies lower as L_x and L_y grow. Given the stability condition (11), the number of eigenfrequencies that can be accounted for ($M(n)$ and $N(m)$ in (7)) also grows as the plate dimensions grow. In the algorithm, this is implemented by adding zero values to q^{t+1}_{mn}, q^{t}_{mn} and q^{t-1}_{mn} (the q-vectors) in (9), where the combinations mn were non-existent in the previous update. If the plate decreases in size, the values of the q-vectors for which the combinations mn do not satisfy the stability condition anymore will be removed. Also in (7), the modal shapes change depending on L_x and L_y in (8). As long as the same number of modes is accounted for, the q-vectors will not be affected in (9) as Φ_{mn} changes. Lastly, the radiation damping α_{rad} and factors A, B and C in (9) will be updated as L_x and L_y change.

The thickness of the plate can also be changed. This changes the thermoelastic damping α_{th} in (5), the radiation damping α_{rad} in (6) and the stiffness factor κ^2 in (2) which then changes the eigenfrequencies again. It can be derived that a decrease in thickness will lower the eigenfrequencies.

4. RESULTS

In this section, the results of the implementation will be discussed. They have been informally evaluated by the authors and sound demos have been made available online [1] . Unfortunately, we did not have access to an actual plate reverb, so to in order to tell whether the implementation was successful we compared it to the *PA1 Dynamic Plate Reverb* and *ValhallaPlate*: already existing VA plugins. When compared to these plugins, we tried to put their settings (plate size, decay values, etc.) as close to the values of our implementation as possible. The outputs from the plugin and our implementation were then compared. The reasons for any differences in sound output we could only speculate on, as the plugins were not open-source and internal parameters were thus hidden.

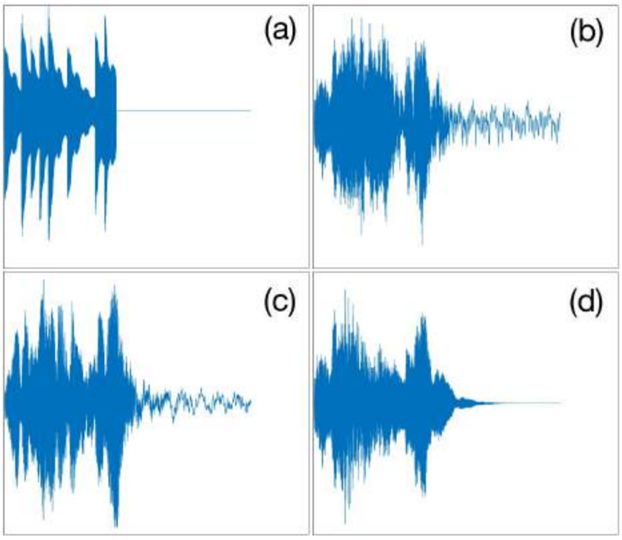


Figure 2. Plots of input and different output examples using EMT140 properties. Total length of outputs: 9.2 seconds. (a): Dry input signal, (b): Left output signal $(0.1L_x, 0.45L_y)$, (c): Left moving output signal ($R_x = R_y = 0.4, S_x = 6, S_y = 5, \theta = 0.5\pi$), (d): Plate stretched from $L_y = 1 - 2$m between $1 - 2$ seconds.

As we did not have any way of comparing our novel additions (moving the outputs and changing plate dimensions) to existing solutions, we could only evaluate the output based on what we expected it to sound like. The waveforms of the input and a few output examples can be found in Figure 2.

4.1 General output

In general, the output sound is very natural, especially when combined with some dry input signal. Although no formal listening tests have been carried out, the naturalness of the has been reported by the authors. When compared to the *PA1 Dynamic Plate Reverb* and *ValhallaPlate* the output has a little more low-frequency content. This is probably because the damping factors we included in our algorithm mostly attenuate the high-frequency content.

4.2 Moving outputs

Moving the outputs creates a result that sounds like a vibrato/flanging effect. This can be explained by the fact that the outputs 'travel away' or 'travel towards' sound that has already been travelling in the plate which changes the pitch of the output sound slightly.

Letting the left and right output move at different speeds or in different shapes causes the sound to arrive in the left channel and the right channel at different time instances. Perceptually, it will sound like the reverb is moving from left to right at different speeds causing a very immersive stereo effect.

4.3 Changing plate dimensions

Increasing the size of the plate creates a pitch-bend effect: lower pitch when L_x and/or L_y are increased and higher

[1] http://tinyurl.com/zwscbtl Full link: [16]

when these are decreased. This can be explained by the fact that the eigenfrequencies are 'stretched/shortened' as the plate dimensions increase/decrease. The exact opposite happens happens when plate thickness is changed: if the plate gets thicker, the pitch goes up, and the other way around. This can be explained by the increase in the stiffness factor in (2) when the thickness increases.

An interesting effect occurs when input and output are combined (not 100% wet signal) as the pitch bend effect only applies to the reverb, and does not influence the dry input signal.

4.4 Dynamic loss coefficients

To the best of our knowledge, the current state of the art does not make use of loss coefficients dependent on physical parameters. We explored the possibility to make some of these parameters dynamic. The most interesting parameter we explored is the air density (ρ_a) in Equation (6). When increasing this, the sound (especially the high frequency content) will die out sooner and has a 'muffled' sound and can also be derived from the equation. Ultimately, we decided against implementing this feature, as it did not add much to the sound; it only decreased the naturalness of the plate reverb.

4.5 Computational speed

As stated in before, the computational speed depends on the total number of eigenfrequencies ($M(n)$ and $N(m)$) being accounted for in the algorithm. We found a correlation of 0.996 between this number and program speed, making decreasing the total number of eigenfrequencies our main focus. What the authors explain in [13] is the possibility of removing the eigenfrequencies that are not perceptually important. In our algorithm we propose:

$$d = (\sqrt[12]{2}^{C/100} - 1) \cdot f_\mathrm{c}, \tag{17}$$

where C is an arbitrary amount in cents and f_c is the current eigenfrequency (in Hz) starting with the lowest eigenfrequency accounted for in the algorithm which can be calculated using $f = \frac{\omega}{2\pi}$. The algorithm will then discard the eigenfrequencies if:

$$f_i - f_\mathrm{c} < d,$$

otherwise:

$$f_\mathrm{c} = f_i.$$

When C is put to 0.1 cents, the total number of eigenfrequencies accounted for (using the EMT140 properties) decreases from 18,218 to 7,932, which makes a big difference in computational time. Now, our implementation only needs only needs roughly 7 seconds instead of 12 seconds to process 9.2 seconds of audio, proving that a real-time implementation would indeed be feasible. The algorithm has been tested using a MacBook Pro containing a 2,2GHz Intel Core i7 processor. The authors indeed report no significant audible difference after reducing the number of modes this way.

In order to improve computational time when the outputs are moving, $\Phi_{mn}(x_\mathrm{out}, y_\mathrm{out})$ is evaluated for different values of t: $t \in [1, f_\mathrm{s}]$ with steps of $4/max([L_x\, L_y])$, before the update equation (9) and is then selected (instead of evaluated) at the appropriate times in the update equation. We found this step-size to be a good balance between speed and having minimal artefacts in the sound if the speed is not set to be too high.

Changing the plate dimensions greatly influences the computational time as the eigenfrequencies need to be recalculated many time. To improve computational time, the eigenfrequencies and the modal shapes are reevaluated every 100th time-step, according to the current (new) dimensions of the plate. As with the previous chosen time-step, found this to be a good balance between speed and having minimal artefacts.

5. CONCLUSION AND FUTURE WORK

In this paper we presented a VA simulation of plate reverberation. The output of the implementation sounds natural and parameters like in- and output positions and plate dimensions have been made dynamic resulting in a very unique and interesting flanging and pitch-bend effect (respectively), something which has not yet been achieved by the current state of the art. Another novelty is that we included thermoelastic and radiation damping that influence every eigenfrequency independently.

In the future we would like to create a real-time plugin containing all that is presented in this paper. In order to do so, the algorithm needs to be optimised, especially computationally heavy processes like moving the pickups and changing the dimensions of the plate. When this is achieved, the plugin will be tested with musicians in order to test usability and whether the output sound is satisfactory. Furthermore, we would like to add damping induced by a porous medium to our implementation. Even though it has been ignored in this work, it is an important feature in the EMT140. Lastly, we would like to explore the possibilities of changing other parameters of the plate using the presented model as a basis. The shape and the structure of the plate, for example, would be very interesting to make dynamic. Moreover, different materials, such as gold and aluminium, or even non-metallic materials like glass, could be explored and result in some interesting timbres.

6. REFERENCES

[1] V. Välimäki, F. Fontana, J. O. Smith, and U. Zölzer, "Introduction to the special issue on virtual analog audio effects and musical instruments," in *IEEE Transactions on Audio, Speech, and Language Processing, VOL. 18, NO. 4*, New York, USA, 2010.

[2] U. Audio. (2010) EMT140 plate reverb powered plug-in for UAD-2 (video). [Online]. Available: https://www.youtube.com/watch?v=_KnMG_OW5c0

[3] J. Menhorn. (2012) EMT140 plate reverb. [Online]. Available: http://designingsound.org/2012/12/emt-140-plate-reverb/

[4] J. M. Jot and A. Chaigne, “Digital delay networks for designing artificial reverberators,” in *Proc. 90th Convention Audio Eng. Soc, preprint 3030*, Paris, France, 1991.

[5] J. S. Abel, D. P. Berners, and A. Greenblatt, “An emulation of the EMT140 plate reverberator using a hybrid reverberator structure,” in *127th AES Convention*, New York, USA, 2009.

[6] V. Välimäki, S. Bilbao, J. O. Smith, J. S. Abel, J. Pakarinen, and D. Berners, *DAFx: Digital Audio Effects Second Edition*. John Wiley and Sons Ltd., 2011.

[7] S. Bilbao and M. van Walstijn, “A finite difference scheme for plate synthesis,” in *Proceedings of the International Computer Music Conference. pp. 119-122*, Barcelona, Spain, 2005, pp. 119–122.

[8] C. Webb. (2016) PA1 dynamic plate reverb. [Online]. Available: http://www.physicalaudio.co.uk/PA1.html

[9] ValhallaDSP. (2015) Valhallaplate. [Online]. Available: http://valhalladsp.com/shop/reverb/valhalla-plate/

[10] K. Arcas, “Physical modelling and measurements of plate reverberation,” in *19th International Congress on Acoustics Madrid*, Madrid, Spain, 2007.

[11] K. Arcas and A. Chainge, “On the quality of plate reverberation,” in *Applied Acoustics 71*, Palaiseau, France, 2010, pp. 147–156.

[12] N. H. Fletcher and T. D. Rossing, *The Physics of Musical Instruments Second Edition*. Springer, 1998.

[13] M. Ducceschi and C. J. Webb, “Plate reverberation: Towards the development of a real-time physical model for the working musician,” in *Proc. of the 22nd International Congress on Acoustics*, Buenos Aires, Argentina, 2016.

[14] T. Irvine. (2013) Response of a rectangular plate to base exitation, revision e. [Online]. Available: http://www.vibrationdata.com/tutorials2/plate_base_excitation.pdf

[15] D. Reinfurt. (2014) Lissajous patterns. [Online]. Available: http://1000things.org/media/image/large/1lissajous.gif

[16] S. Willemsen. (2017) Sound files. [Online]. Available: https://www.dropbox.com/s/wy2ceyklx3wptu8/Sound%20Files%20VA%20Plate%20Reverb.zip?dl=0

The Effectiveness of Two Audiovisual Mappings to Control a Concatenative Synthesiser

Augoustinos Tsiros, Grégory Leplâtre
Centre for Interaction Design
Edinburgh Napier University
10 Colinton Road, EH10 5DT
a.tsiros@napier.ac.uk,g.leplatre@napier.ac.uk

ABSTRACT

This paper presents the results of an investigation into audio-visual (AV) correspondences conducted as part of the development of Morpheme, a painting interface to control a corpus-based concatenative sound synthesiser. Our goal was to measure the effects of the corpora on the effectiveness of the AV mappings used by Morpheme to synthesise sounds from images. Two mapping strategies and four corpora were empirically evaluated by 110 participants. To test the effectiveness of the mappings, sounds were generated using Morpheme from images designed to exhibit specific properties. The sounds were then played to the participants of the study and for each sound, they were presented with three images: the image used to synthesise the sound and two similar distractor images. Participants were asked to identify the correct source image and rate their level of confidence. The results show that the audio corpus used to synthesise the audio stimuli had a significant effect on the subjects' ability to identify the correct image while the mapping didn’t. No effect of musical/sound training was observed.

1. INTRODUCTION

In recent years, there has been a growing interest in the study of multisensory associations, in particular between audio and visual stimuli [1]–[7]. Previous research suggests that the mapping between the user sensorimotor inputs to sound synthesis and musical parameters can affect a number of cognitive and experiential aspects of interaction with sound synthesis as well as the expressivity that derives from the interaction [8], [9]. This understanding has led artists and scientists to investigate cross-sensory mapping strategies for musical interaction.

The research presented in this paper is motivated by the development of *Morpheme*, an interactive application that allows users to synthesise sounds through sketching. The purpose of the system is to facilitate the expression of sound design ideas by describing the qualities of the sound to be synthesised in visual terms. For Morpheme to work effectively, its design must be based on meaningful relationships between what the user paints and what they hear. Sounds are generated in Morpheme using corpus-based concatenative synthesis [10].

Previous studies that investigated correspondences between the different senses have shown that there is consistency in the feature correlates which subjects perceive as a good match. For instance, listeners have consistently reported high levels of perceived correspondence between auditory pitch and light intensity (higher auditory pitches for higher luminosity) [11]–[16], visual size and pitch (lower pitches for bigger visual shapes), visual size to loudness (louder sounds for bigger visual shapes) [17]–[22], spatial height to pitch (higher pitches for higher spatial location) [12], [14], [16], [23]–[30]. Some studies have reported a high correlation between texture granularity and sound compactness (high visual granularity for dense audio spectrum) [31], [32] and visual repetitiveness and sound dissonance (harmonic sound for repetitive textures) [19], [31], [32]. While these results can be applied to design of an empirically validated mapping to convert the user’s sketches into sound, some important questions remain regarding the effectiveness of such AV mappings in a real-world application.

Recent findings suggest that interactions between different dimensions of auditory and visual objects in multidimensional contexts can occur. For example, an increase in tempo rate is often correlated to an increase in visual motion speed, however when an increase in auditory tempo is accompanied by a decreasing loudness, the correspondence between tempo and speed is weakened [33]. These types of interactions between auditory and visual parameters could potentially compromise the direct application of AV associations that have only been tested independently. Mappings between users’ sensorimotor input and sound and musical parameters commonly consist of more than one dimension. For instance, *Morpheme*’s mapping associates five auditory and visual features. A question then arises: are there interactions between the AV associations that the mapping consists of?

In addition, the strength of an association also depends on the dynamism of the parameters involved [33], [34]. For instance, higher pitch sounds are usually perceived as congruent with smaller visual size. However, ascending pitch is congruent with a growing size and descending pitch with shrinking size [34]. This suggests that the congruence strength can be affected and even reversed depending on whether the stimuli are dynamic or static.

Furthermore, many descriptors deriving from various types of audio analysis are available for use in a potential AV mapping, likewise for the descriptors of an image. This raises the question of which descriptors will be most effective in AV mappings. Finally, the auditory stimuli in most of the studies commonly undertaken tend to be limited to simple synthetic tones. Considering that most natural and musical sounds are more complex than pure sine tones whether these results still apply to complex real-world sounds is uncertain. These are important questions to answer for the development of Morpheme as it uses a multi-dimensional mapping and concatenative sound synthesis uses pre-recorded audio material to synthesise sound that can have complex timbre characteristics.

2. MORPHEME

Morpheme is an application that allows sound synthesis to be controlled by sketching. The underlying synthesis technique used is concatenative synthesis. For more information about the graphical user interface of the system, please see [35]. Morpheme extracts visual features from the analysis of the sketch drawn and uses these features to query audio units from Catart's database [10]. During playback, windowed analysis is performed on the pixels of the sketch. A window scans the sketch from left to right one pixel at every clock cycle, the rate of which is determined by the user. Only the areas of the canvas that are within the boundaries of the window area are analysed. The window dimensions can be determined by the user, however the default size of the analysis window is 9 pixel wide by 240 pixel height. For more detailed information about the implementation, see [36].

2.1 The AV Mapping

Some of the most effective empirically validated audio-visual associations discussed in the introduction were selected to develop two mappings that enable the selection of audio units from the corpus. One of the main distinctions between the mappings is that one is *achromatic* while the other is *chromatic*. The first mapping is considered achromatic because the visual descriptors extracted from *Morpheme's* sketch do not use any colour information, whereas the second mapping does. **Table 1** and **Table 2** show the details of the two mappings.

Table 1. Achromatic mapping associations between audio and visual descriptors for the retrieval of audio.

Visual Features	Audio Features
Texture granularity	Spectral flatness
Vertical position	Pitch
Texture repetitiveness	Periodicity
Size	Loudness
Horizontal length	Duration

Table 2. Chromatic mapping associations between audio and visual descriptors for the retrieval of audio.

Visual Features	Audio Features
Color variance	Spectral flatness
Color brightness	Spectral centroid
Brightness variance	Periodicity
Size	Loudness
Horizontal length	Duration

3. METHOD

An experiment was conducted to answer the following questions

- Which overall mapping (chromatic/achromatic) is the most effective?
- Do the sounds used as the source audio (corpus) by the concatenative synthesiser have an effect the effectiveness of the mappings?
- Is the effectiveness of the mappings affected by the experience of the respondents with sound/music?

3.1 Subjects

A total of 110 volunteers took part in the study. The study was conducted in two conditions, in what will be referred to as the *supervised* and *unsupervised* conditions. The supervised study took place in Edinburgh Napier University's Auralization suite. Participants used Beyer Dynamics DT 770 Pro monitoring headphones to listen to the audio stimuli. A 17.3 inch laptop screen was used for viewing the visual stimuli. SurveyGismo was used for viewing the stimuli and for recording the participants' responses. The unsupervised study ran as an online survey. In the supervised study a total of 44 participants volunteered to take part. Sixteen were sound practitioners (referred to as experts), and twenty-eight non-sound practitioners (referred to as non-experts). All of the expert participants except two played a musical instrument and the self-reported levels of expertise were 4 basic, 6 intermediate and 4 advanced.

In the unsupervised study, participants were similarly divided into two groups. There were a total of 66 participants recruited via a post on audio and music related mailing lists. The expert group consisted of 39 participants and the non-expert group of 27. All of the expert participants except one played a musical instrument and the self-reported levels of expertise were 11 basic, 14 intermediate and 14 advanced. Twenty eight of the expert participants had received formal music theory training for at least one year. All of the participants reported using analogue and digital equipment for sound synthesis, signal processing, sequencing and audio programming tools.

3.2 Stimuli

3.2.1 Audio Stimuli

Four audio stimuli were tested for each mapping, i.e. four sounds per mapping, resulting in a total of eight audio stimuli. Each sound was synthesized using one of the four corpora (i.e. String, Wind, Impacts, Birds). The resulting sounds used as stimuli had a duration of 8 seconds each. The string and the birds' corpora consisted of mainly harmonic and periodic sounds, while the wind and the impacts corpora consisted of mainly unpitched, non-harmonic sounds. The string and the wind corpora consisted of a small number of audio-units that were very homogenous (i.e. all audio-units had very similar characteristics (loudness, periodicity, pitch)), while the impacts and the bird corpora consisted of audio-units that were non- homogeneous (i.e. more variation in terms of loudness, periodicity and frequency contain).

3.2.2 Visual Stimuli

Six visual stimuli were designed for this experiment. The aim was to test all the associations used in the mapping and to manipulate multiple variable simultaneously. To further validate the effectiveness of the mapping the three images presented for each audio stimuli have been designed to be similar and vary only in some respects (see **Figure 1**). For instance, the first and second images in **Figure 1** are more similar in terms of pitch contour, while the first and third images are similar in terms of texture granularity. This decision aimed to increase the level of difficulty of the task by introducing a confusion variable.

3.3 Procedure and Study Design

Participants were presented with a sequence of short sounds synthesised as described in the previous section. They were given a selection of three images with each sound. Participants were told during the training session that only one of the three images displayed on the screen had been used to generate each sound they were presented with. By clicking on an on-screen button using a computer mouse, subjects could playback the audio files as many times as they liked. After listening to each individual sound, participants could use an on-screen radio button to pick one of the three images they thought had been used to generate the sound using Morpheme. After each choice, they were asked to rate their level of confidence in their judgement. The series of tasks took approximately five minutes. The order of presentation of the sounds was randomized.

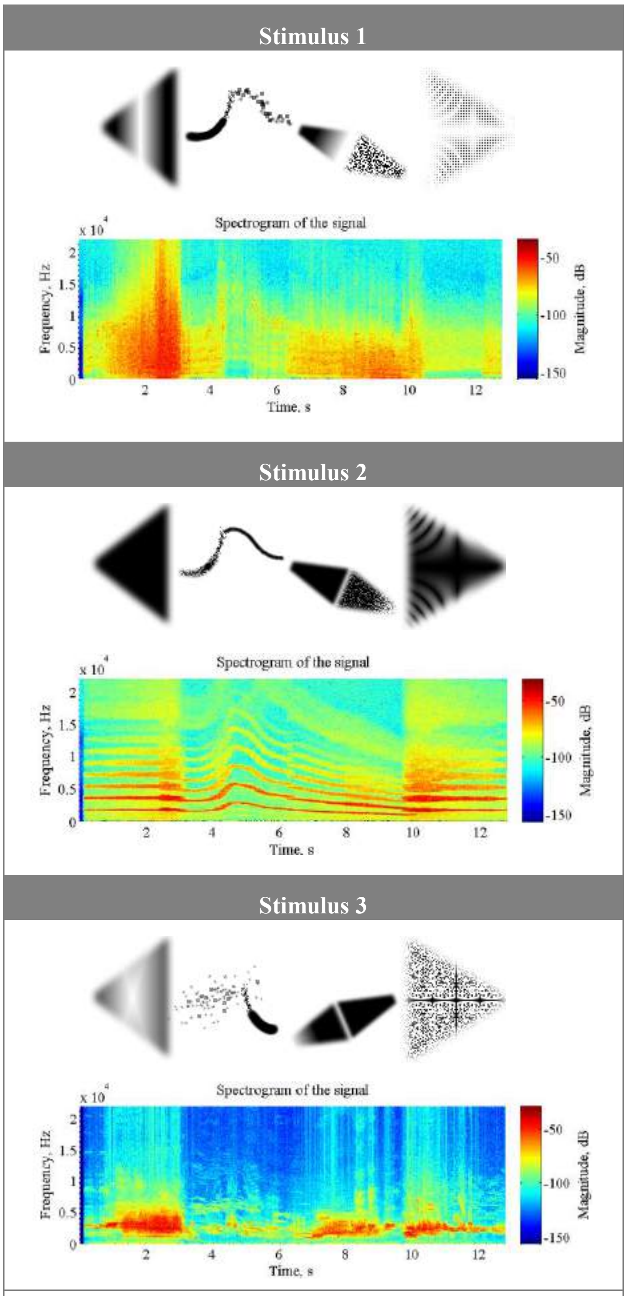


Figure 1. Example of visual and audio stimuli used for testing the Achromatic Mapping.

4. RESULTS

4.1 Correct Image Selection Rate

This section presents the results of the image selection tasks. **Figure 2** and **Figure 3** show the percentage of correct image selection made. Pearson's Chi-square tests of independence were performed to test the null hypothesis of no association between the factors (i) subjects' correct image selection rate and the audio corpora, and (ii) subjects' correct image selection rate and AV mappings.

A strong relationship between correct selection rate and the audio corpora X^2*(3, N = 224)* = 19.22, *p* = .0 was observed. No significant relationship was noted between correct selection rate and the mapping $X^2(1, N = 224) = 1.143$, $p = .28$.

For the expert group in the supervised condition, data analysis shows a strong relationship between the correct selection rate and the audio corpora $X^2(3, N = 128) = 9.86$, $p = .02$. No significant relationship was noted between correct selection rate and the mapping $X^2(1, N = 128) = 3.33$, $p = .068$.

For the non-expert group in the unsupervised condition, data analysis shows that there is a strong relationship between correct selection rate and the audio corpora $X^2(3, N = 216) = 11.37$, $p = .010$. No significant relationship was revealed between correct selection rate and the mappings $X^2(1, N = 216) = .70$, $p = .40$.

For the expert group in the unsupervised condition, data analysis shows that there is a strong relationship between the ability of the subjects to identify the correct image and the audio corpora $X^2(3, N = 312) = 23.15$, $p < .001$. Again, no significant relationship was observed between the correct selection rate and the mappings $X^2(1, N = 312) = .3.07$, $p = .079$.

Post-hoc testing of each group/condition chi-square results was performed for each level of the factor corpus in order to determine which factor levels contributed to the statistical significance that was observed. The results of the post-hoc test are presented in **Table 3**. For more information about the approach followed to perform post-hoc testing refer to [37] and [38].

4.2 Comparison between expert & non-expert groups

This section presents the results obtained from the comparison between the expert and the non-expert groups in the supervised and unsupervised condition. A Pearson's chi-square test of independence was performed on three between subject variables to test the null hypothesis of no association between: (i) the subjects' skills and correct image selection rate, (ii) subjects' skills and correct image selection rates difference between the two mappings, and (iii) subjects' skills and correct image selection rates difference between the four corpora.

The comparison between the expert and the non-expert groups in the supervised condition shows that there is an association between the variables subjects skills and correct selection rate $X^2(1, N = 352) = 4.43$, $p = .035$.

Further testing of the relationship between the variables correct detection and skills was performed by layering the participant responses based on the two levels of the mapping factor. The results of the test show that the association between the subjects' skills and correct selection rate was stronger when the chromatic mapping was tested $X^2(1, N = 176) = 4.27$, $p = .039$ and not strong when the achromatic mapping was tested $X^2(1, N = 176) = .88$, $p = .347$.

The comparison between the expert and the non-expert groups in the unsupervised condition show no association between the variables subjects skills and correct selection rate $X^2(1, N = 528) = 0.10$, $p = .747$. Further testing of the relationship between the variables correct detection and skills was performed by layering the participant responses based on the two levels of the mapping factor. The results of the test show that there was no association between the subjects' skills and correct selection rate for either of the mappings: achromatic $X^2(1, N = 264) = .011$, $p = .917$; chromatic mapping $X^2(1, N = 264) = .332$, $p = .564$. Post-hoc testing showed that the relationship between correct detection and subjects' skills in the different experiment conditions was not significant, see **Table 4.**

4.3 Confidence Ratings

This section presents the results obtained from the second question of the experiment, were expert and non-expert subjects in the supervised and unsupervised conditions reported on their confidence levels regarding their decision (i.e. corresponding image for each audio stimuli). A repeated measure ANOVA using a general linear model was computed with the confidence ratings as within subjects factor and the corpora and the mapping as between subjects factors. The analysis aimed to test the null hypothesis of no association between the subjects' confidence ratings and the mappings, or the audio corpora.

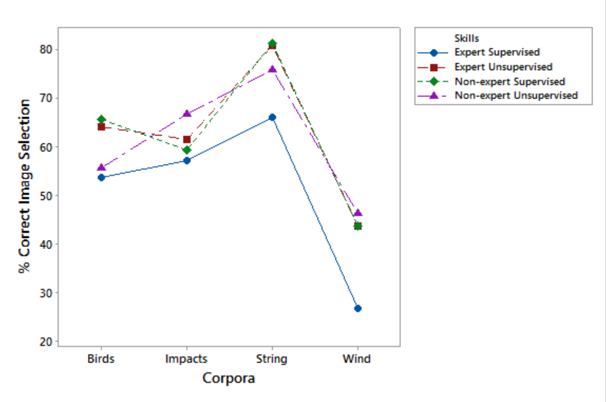


Figure 2. Overall effect of the corpus on the subject's ability to identify the correct image.

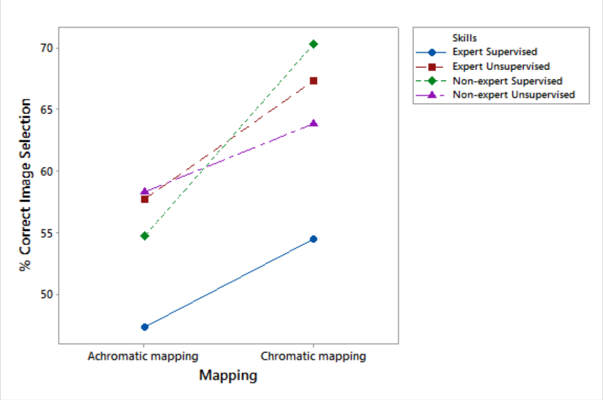


Figure 3. Overall effect of the mapping on the subject's ability to detect the correct stimuli.

Table 3. Post-hoc testing of each group/condition chi-square results by estimating p values from the chi-square residuals for each level of a factor and comparing these to an adjusted benferonni corrected p value. An asterisk (*) indicates statistical significance.

	Experts (s)		**Experts (u)**		**Non-experts(s)**		**Non-experts(u)**	
Corpus	X^2	**p**	X^2	**p**	X^2	**p**	X^2	**p**
Birds	0.17	.67	0.11	.73	0.2	.64	0.93	.33
Impacts	0.17	.67	0.04	.83	1.1	.28	0.93	.33
String	6.3	.01	14.8*	<.006	6.8	.008	6.6	.009
Wind	6.3	.01	15.8*	<.006	17.3*	<.006	6.6	.009
N	32		78		56		54	
Total	$X^2(3, N=128) = 9.8, p < .05$		$X^2(3, N=312) = 23.1, p < .01$		$X^2(3, N=224) = 19.2, p < .01$		$X^2(3, N=226) = 11.3, p < .05$	

s=supervised / u=unsupervised
benferroni adjusted p value 0.0062

Table 4. Post-hoc testing of correct detection chi-square results for subjects' skills factor.

Skills	X^2	**p**	**N**
Expert (s)	.6	.41	128
Expert (u)	2.1	.14	312
Non-Expert (s)	8.5	.033	224
Non-Expert (u)	.43	.51	216
Total	$X^2(3, N = 880) = 8.711, p = .033$		

s=supervised/u=unsupervised
Benferroni adjusted p value 0.00625

Analysis of the subjects' confidence ratings revealed no significant differences due to the corpus with an exception for the experts in the unsupervised condition, see **Table 5**. Significant differences were noted between each mapping for the means of the expert group. No interactions between the factors mapping and audio corpus were revealed. The results of the analyses are shown in **Figure 4**. An additional ANOVA model was computed to examine the relationship between reported confidence and correct image identification which was found to be significant for all the groups with only the exception of the expert group in the supervised condition. The results of this analysis are shown in **Table 6**.

4.4 Summary of Results

The answer to the first question of this study is that the choice of mapping did not affect participants' image selection success. A trend that can be observed in the data indicates that participants selected the correct images well above chance levels, regardless of the mapping.

The answer to the second question of the study is that the audio corpus used to synthesise the audio stimuli from the images had a significant effect on the subjects' ability to identify the correct image. The image selection success rate was higher when the string corpus was used, followed by the birds and the impact corpus, while the lowest success rate was observed for the wind corpus. The analysis showed that the corpora that had the strongest effect on the subjects' successful selection rate were the string and the wind.

In response to the third question, there was no statistically significant difference between the results of the expert and non-expert groups.

Table 5. General ANOVA model computed to investigate the effect of the AV associations, corpus and mapping on the perceived similarity reported by the subjects. An asterisk (*) indicates statistical significance.

	Corpus		**Mapping**		**Interactions**		
Group	***F***	***P***	***F***	***P***	***F***	***P***	***Total***
Non-experts (s)	2.25	.1	.9	.4	1.89	.1	223
Non-experts (u)	.28	.8	14.14*	<.001	1.93	.1	215
Experts(s)	.70	.5	2.26	.1	0.75	.5	127
Experts(u)	4.74*	<.005	15.51*	<.001	1.7	.1	311
df		3	1		1		

s=supervised / u=unsupervised

Table 6. General ANOVA model computed to investigate the effect of correct detection on the reported confidence level. An asterisk (*) indicates statistical significance.

	Correct selection		
Group	***F***	***P***	***Total***
Non-experts (s)	7.53*	<.005	223
Non-experts (u)	6.15*	<.05	215
Experts(s)	.34	.5	127
Experts(u)	16.73*	<.001	311
df	3		

s=supervised / u=unsupervised

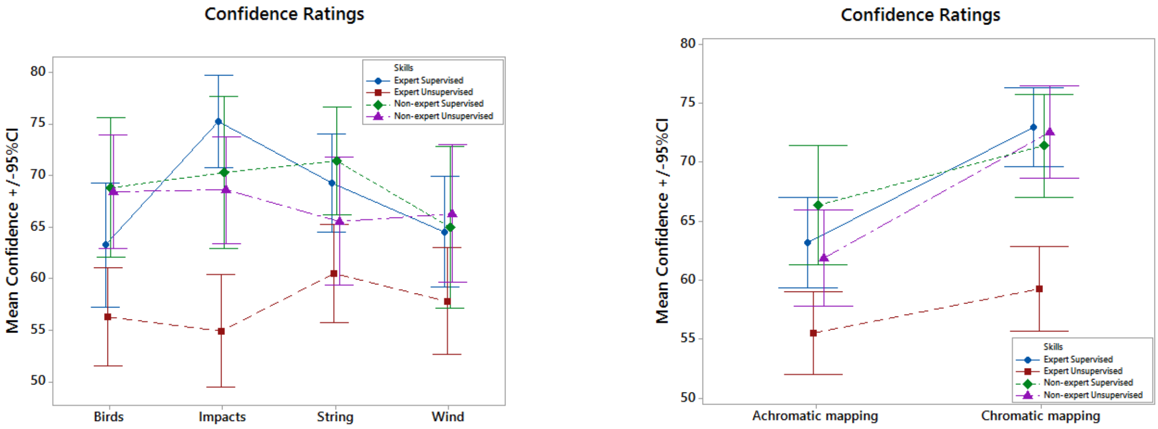


Figure 4. Data means and confidence intervals of participants' confidence ratings for the mapping and corpus factors for all of the groups. If intervals do not overlap the corresponding means are statistically significant.

5. DISCUSSION

It is worth noting that overall there are consistencies in the subjects' responses across the groups and conditions. The fact that there are consistencies between the two subject groups (i.e. experts, non-experts) suggests that musical/sound training was not important. Furthermore consistencies in the subjects' responses between the two experimental conditions (controlled environment and online) suggest that both approaches to gathering data are equally well suited to this type of experiment.

The first result of the study is that the choice of audio corpus had an effect on participants' image selection success rate and therefore affects the effectiveness of the mappings. A first interpretation is that when the sound corpus is not harmonic and continuous, the resulting sounds can be noisy and lack clarity. This could affect the effectiveness of the mapping by causing a reduction in the salience of the audio-visual associations. This in turn weakens the ability of the participants to pay attention to the causal relationship between the image and the sound.

As stated in Section 4.4, there was no statistically significant differences between expert and non-expert subjects. The fact that musical/sound training was not a very significant factor suggests that the correspondences might be underpinned by either similarities between auditory and visual perception and/or by cultural conventions, but not specifically by the acquisition of advanced musical and listening skills. These findings are in agreement with [18] and oppose the findings of [17], [20], [39].

Finally, the participants' confidence ratings revealed no correlation between confidence levels and correct image selection. Although a trend can be observed between correct image selection and confidence levels reported by subjects (i.e. confidence levels on average are higher when participants have responded correctly rather than incorrectly), the confidence levels reported on incorrect responses are very high (i.e. in most cases well above 50%), which does not indicate that participants were aware that their responses were incorrect. Overall, participants felt more confident when the chromatic mapping and the string corpus were used. Furthermore, the confidence ratings show that when the wind corpus was used, participants felt least confident, however for the other three corpora the results were not following a strong correlation pattern.

6. CONCLUSIONS

We reported on the evaluation of Morpheme's Audio-Visual mappings used to control a corpus-based concatenative synthesizer through painting. One of the main motivations of the present investigation was to explore how to best utilise empirically validated audio-visual associations in the design of sound synthesis user interfaces. Many studies have shown that people often map stimulus properties from different sensory modalities onto each other in a manner that is surprisingly consistent. Furthermore, previous research suggests that correspondences of stimulus properties may be innate or learned at very early stages of a person's life, [40]–[42]. This research project is motivated by the idea that mappings which are consistent with users' prior perceptual knowledge and the subtleties of multisensory aspects of perception, can better accommodate their sensorimotor skills and support the comprehension of information and interaction with computer systems.

The present experiment tested the effects of four concatenative synthesiser corpora on the effectiveness of two AV mappings, measured by the ability of subjects to identify the image used to synthesise a sound they were presented with. The results showed that when multiple parameters vary simultaneously, the nature of the sounds present in the audio corpus has a moderating effect on the mapping effectiveness. For example, when the corpus consisted of harmonic sounds, participants' detection rates significantly increased in comparison to when using inharmonic sounds. Finally, results indicated that sound/musical training had no significant effect on the perceived similarity between AV features or the discrimination ability of the subjects.

Further work includes making improvements to the mapping of Morpheme, for instance to minimise interactions between audio parameters used (e.g. spectral flatness and periodicity). A set of less correlated audio features is likely to lead to more effective mappings. Another direction for future work involves testing a wider range of audio-visual features.

Acknowledgments
The authors would like to acknowledge the Institute for Informatics & Digital Innovation for their financial support and IMTR research centre for sharing the *CataRT* system.

2 REFERENCES

[1] B. Caramiaux, J. Francoise, F. Bevilacqua, and N. Schnell., “Mapping Through Listening,” in *Computer Music Journal*, vol. 38, no. 2006, 2014, pp. 1–30.

[2] B. Caramiaux, F. Bevilacqua, and N. Schnell, “Towards a gesture-sound cross-modal analysis,” in *Gesture in Embodied Communication and Human-Computer Interaction*, I. Kopp, S., Wachsmuth, Ed. Springer, Heidelberg, 2010, pp. 158–170.

[3] M. Leman, *Embodied Music Cognition and Mediation Technology*. MIT press, 2008.

[4] R. I. Godøy, “Images of Sonic Objects,” *Organised Sound*, vol. 15, no. 1, p. 54, Mar. 2010.

[5] D. Smalley, “Spectromorphology: explaining sound-shapes,” *Organised sound*, vol. 2, pp. 107–126, 1997.

[6] T. Wishart, *On Sonic Art*. Hardwood academic publishers, 1996.

[7] M. Blackburn, “The Visual Sound-Shapes of Spectromorphology: an illustrative guide to composition,” *Organised Sound*, vol. 16, no. 1, pp. 5–13, Feb. 2011.

[8] a. Hunt and R. Kirk, “Radical user interfaces for real-time control,” *Proc. 25th EUROMICRO Conf. Informatics Theory Pract. New Millenn.*, pp. 6–12 vol.2, 1999.

[9] A. Hunt, M. Wanderley, and R. Kirk, “Towards a Model for Instrumental Mapping in Expert Musical Interaction,” in *International Computer Music Conference*, 2000.

[10] D. Schwarz, G. Beller, B. Verbrugghe, and S. Britton, “Real-Time Corpus-Based Concatenative Synthesis with CataRT,” in *Digital Audio FX*, 2006, pp. 279–282.

[11] L. E. Marks, “On associations of light and sound: The mediation of brightness,” *J. Psychol.*, vol. 87, pp. 173–188, 1983.

[12] I. H. Bernstein and B. A. Edelstein, “Effects of some variations in auditory input upon visual choice reaction time.,” *J. Exp. Psychol.*, vol. 87, pp. 241–247, 1971.

[13] T. L. Hubbard, “Synesthesia-like mappings of lightness, pitch, and melodic interval,” *Am. J. Psychol.*, vol. 109, no. 2, pp. 219–238, 1996.

[14] R. D. Melara and T. P. O’Brien, “Interaction between synesthetically corresponding dimensions.,” *J. Exp. Psychol.*, vol. 116, pp. 323–336, 1987.

[15] R. D. Melara, “Dimensional interaction between color and pitch.,” *J. Exp. Psychol. Hum. Percept. Perform.*, vol. 15, pp. 69–79, 1989.

[16] G. R. Patching and P. T. Quinlan, “Garner and congruence effects in the speeded classification of bimodal signals.,” *J. Exp. Psychol. Hum. Percept. Perform.*, vol. 28, pp. 755–775, 2002.

[17] M. B. Küssner, “Shape, drawing and gesture: cross-modal mappings of sound and music,” King’s College London, 2014.

[18] S. Lipscomb and E. Kim, “Perceived match between visual parameters and auditory correlates: An experimental multimedia investigation,” in *International Conference on Music Perception and Cognition*, 2004, pp. 72–75.

[19] F. Berthaut, M. Desainte-catherine, and M. Hachet, “Combining audiovisual mappings for 3d musical interaction,” *International. Computer Music Conference*, 2010.

[20] R. Walker, “The effects of culture , environment , age , and musical training on choices of visual,” vol. 42, no. 5, pp. 491–502, 1987.

[21] L. B. Smith and M. D. Sera, “A Developmental Analysis of the Polar Structure of Dimensions,” *Cogn. Psychol.*, vol. 142, pp. 99–142, 1992.

[22] C. Spence, “Crossmodal correspondences: a tutorial review.,” *Atten. Percept. Psychophys.*, vol. 73, no. 4, pp. 971–95, May 2011.

[23] E. Ben-Artzi and L. E. Marks, “Visual-auditory interaction in speeded classification: Role of stimulus difference,” *Percept. Psychophys.*, vol. 57, no. 8, pp. 1151–1162, Nov. 1995.

[24] Z. Eitan and R. Timmers, “Beethoven’s last piano sonata and those who follow crocodiles: cross-domain mappings of auditory pitch in a musical context.,” *Cognition*, vol. 114, no. 3, pp. 405–22, 2010.

[25] K. Evans and A. Treisman, “Natural cross-modal mappings between visual and auditory features,” *J. Vis.*, vol. 10, pp. 1–12, 2010.

[26] E. Rusconi, B. Kwan, B. L. Giordano, C. Umiltà, and B. Butterworth, “Spatial representation of pitch height: the SMARC effect.,” *Cognition*, vol. 99, no. 2, pp. 113–29, 2006.

[27] S. Hidaka, W. Teramoto, M. Keetels, and J. Vroomen, “Effect of pitch-space correspondence on sound-induced visual motion perception.,” *Exp. brain Res.*, vol. 231, no. 1, pp. 117–26, Nov. 2013.

[28] L. E. Marks and Z. Eitan, “Garner’s paradigm and audiovisual correspondence in dynamic stimuli: Pitch and vertical direction,” *Seeing Perceiving*, vol. 25, pp. 70–70, Jan. 2012.

[29] L. E. Marks, “On Cross-Modal Similarity : The

Perceptual Structure of Pitch , Loudness , and Brightness," *J. Exp. Psychol. Hum. Percept. Perform.*, vol. 15, no. 3, pp. 586–602, 1989.

[30] R. Chiou and A. N. Rich, "Cross-modality correspondence between pitch and spatial location modulates attentional orienting.," *Perception*, vol. 41, pp. 339–353, 2012.

[31] K. Giannakis, "Sound Mosaics," Middlesex University, 2001.

[32] K. Giannakis, "A comparative evaluation of auditory-visual mappings for sound visualisation," *Organised Sound*, vol. 11, no. 3, pp. 297–307, 2006.

[33] Z. Eitan, "How pitch and loudness shape musical space and motion: new findings and persisting questions," in *The Psychology of Music in Multimedia*, S.-L. Tan, A. J. Cohen, S. D. Lipscomb, and R. A. Kendall, Eds. Oxford University Press, 2013, pp. 161–187.

[34] Z. Eitan, A. Schupak, A. Gotler, and L. E. Marks, "Lower pitch is larger, yet falling pitches shrink: Interaction of pitch change and size change in speeded discrimination," *Proc. Fechner Day*, 2011.

[35] A. Tsiros and G. Leplâtre, "Evaluation of a Sketching Interface to Control a Concatenative Synthesiser," in *42th International Computer Music Conference*, 2016, pp. 1–5.

[36] A. Tsiros, "A Multidimensional Sketching Interface for Visual Interaction with Corpus-Based Concatenative Sound Synthesis," Edinburgh Napier University, 2016.

[37] T. M. Beasley and R. E. Schumacker, "Multiple Regression Approach to Analyzing Contingency Tables: Post Hoc and Planned Comparison Procedures," *Journal of Experimental Education*, vol. 64, no. 1. pp. 79–93, 1995.

[38] M. a. Garcia-perez and V. Nunez-anton, "Cellwise Residual Analysis in Two-Way Contingency Tables," *Educ. Psychol. Meas.*, vol. 63, no. 5, pp. 825–839, 2003.

[39] M. B. Kussner and D. Leech-Wilkinson, "Investigating the influence of musical training on cross-modal correspondences and sensorimotor skills in a real-time drawing paradigm," *Psychol. Music*, vol. 42, no. 3, pp. 448–469, Jul. 2013.

[40] S. Jeschonek, S. Pauen, and L. Babocsai, "Cross-modal mapping of visual and acoustic displays in infants: The effect of dynamic and static components," *J. Dev. Psychol.*, vol. 10, pp. 337–358, 2012.

[41] S. Wagner, E. Winner, D. Cicchetti, and H. Gardner, "Metaphorical" mapping in human infants.," *Child Dev.*, vol. 52, pp. 728–731, 1981.

[42] P. Walker, J. G. Bremner, U. Mason, J. Spring, K. Mattock, A. Slater, and S. P. Johnson, "Preverbal infants' sensitivity to synaesthetic cross-modality correspondences.," *Psychol. Sci.*, vol. 21, no. 1, pp. 21–5, Jan. 2010.

Extraction, Mapping, and Evaluation of Expressive Acoustic Features for Adaptive Digital Audio Effects

Jonas Holfelt, Gergely Csapo, Nikolaj Andersson, Sohejl Zabetian,
Michael Castanieto, Daniel Overholt, Sofia Dahl, Cumhur Erkut
Aalborg University Copenhagen
{jholfe13,gcsapo13,na13,szabet16,mcasta16}@student.aau.dk
{dano,sof,cer}@create.aau.dk

ABSTRACT

This paper describes the design and implementation of a real-time adaptive digital audio effect with an emphasis on using expressive audio features that control effect parameters. Research in adaptive digital audio effects is covered along with studies about expressivity and important perceptual sound descriptors for communicating emotions. This project was aiming to exploit sounds as expressive indicators to create novel sound transformations. A test was conducted to see if guitar players could differentiate between an adaptive and non-adaptive version of a digital audio effect. The participants could hear a difference, especially when performing expressively. Although the adaptive effect did not seem to enhance expressive capabilities, participants did report an increased sense of control and general awareness of the effect. Overall, the preference over the two versions varied evenly between participants.

1. INTRODUCTION

Throughout history, advances in technology have played a huge role in the development and progression of music [1]. Digital audio is but another advancement built upon foundations established by the early pioneers of digital processing [2]. Today, the progression of music technology is as relevant as ever, and digital audio is at the heart of a wide range of applications, from music production and performance, to the very way in which music is consumed.

’Adaptive’ effects automatically modify control parameters through mappings based on the analysis and extraction of audio features derived from the audio signal [3]. Traditionally, adaptive effects tend to be based on the dynamic properties of sound (examples of this include the compressor and noise gate) [3]. In this work, however, we will refer to adaptive effects as effects based on spectral properties of the input signal, which have a direct correlation to the notion of musical expression [4]. While adaptive effects can be applied and used for many different purposes, this paper will focus on its use with electric guitar playing.

Expression plays a crucial role in musical performances [5]. Musicians tend to communicate emotions and messages through expressive play, which often involves expressive improvisation and deviations from notated music [6, 7].The expressive intention of a performer can be investigated through the analysis of acoustic tone parameters, as these parameters carry sufficient information to identify some of the underlying emotions that the musician is trying to convey [5].

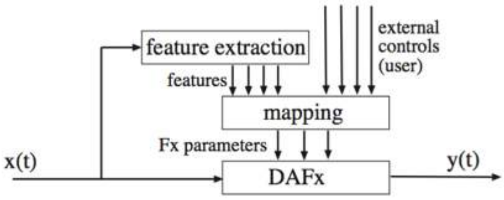


Figure 1. *”Structure of an adaptive DAFx (A-DAFx), with the input signal and the output signal. Features are extracted, and then a mapping is done between these features and user controls as input, and effect parameters as output”.* (Verfaille and Arfib, 2001 [4]).

Temporal variations in timing, dynamics and timbre, all contribute to the expressive quality of musical performances. Such variations in the acoustic signal have been previously considered in the context of modelling expressive performances for analysis [5, 6, 8] and synthesis [3, 7, 9, 10]. However, so far they have not been considered in conjunction with expressive intentions to control adaptive audio effects. An interesting approach would be to exploit such intentions with the purpose of interacting with audio effects in musical performances.

The novelty of this approach has created an opportunity to implement and evaluate a system which extracts expressive acoustical parameters, and maps them to relevant effect control parameters. A multi-effect delay engine was chosen, as it provides scope to clearly test and interpret parameter mapping relationships. In order to create this system, the general structure for adaptive effect implementation was used with the following steps (see Fig 1) [4]:

1. Analysis and feature extraction.
2. Mapping between features and effect parameters environment.
3. Transformation and Re-synthesis

2. RELATED WORK

Research on digital audio effects is highly interdisciplinary in the sense that a wide variety of topics and technical principles have to be understood for their design and implementation. This section will take a methodical approach in considering the various research paths explored during the process of this project.

2.1 Adaptive Digital Audio Effects

Verfaille et al. proposed a new type of audio transformation: 'adaptive digital audio effects' (A-DAFx). This type of effect is also referred to as dynamic processing, intelligent effects, or content based transformations since the effect parameters are altered by the input source over time [4]. The idea of altering the output sound based on the input already exists in commonly used effects such as compressors and auto-tune. However, adaptiveness introduces a new kind of control since specific sound features (or sound descriptors) can be mapped to alter control parameters of the output. The implemented echo effect uses the auto-adaptive approach, which is defined as an effect where acoustic features are extracted from the same source as to which the effect is applied.

The mapping of the sound features is an important aspect of A-DAFx. Several different mapping strategies have been suggested for adaptive effects, covering both sound and gesture inputs [3, 4, 11, 12]. However, only sound features were considered for the purpose of this project. An important consideration for mapping of sound features is the mapping strategy between extraction and manipulation parameters. This choice can be difficult to argue for, since it depends on musical intention [4]. If the mapping gets too complex, the users ability to manipulate the mapping depends on their prior knowledge about DAFx and the techniques used.

2.2 Examples of A-DAFx

At this time, only three commercial adaptive products are known to be aimed at guitarists, these are the 'TE-2 Tera Echo', 'MO-2 Multi Overtone' and 'DA-2 Adaptive Distortion' by 'BOSS' [13–15]. However, no technical descriptions of these effects are known to be available.

2.3 Analysis and Extraction of Expressive Acoustic Features

Adaptive effects are classified based on perceptual attributes, the main perceptual axes in sound being: timing, pitch, timbre, loudness and space, with the latter being the least significant from a musical performance perspective [3]. The first four perceptual attributes will now be examined in the context of expressivity.

The feature extraction portion of the project was to be used for identifying musical expression in the acoustic spectrum of the input signal, and required general insight into various music processing techniques and algorithms [3,11,16]. Acoustic tone parameters constitute of information outlining not only musical structure, but also the expressive intentions of the performer. Friberg et. al stated that *'emotional expression of a music performance can be accurately predicted using a limited set of acoustic tone parameters'* [5].

Studies have found that variation of timing and amplitude are essential features of auditory communication and music, as they carry information about expressive intentions and provide higher resolution of auditory information [6]. In addition to the variety of existing software for extracting tone parameters (e.g. audio to MIDI converters), Friberg et. al developed the algorithm CUEX [5]to identify expressive variations in music performances, rather than providing accurate musical notations. The CUEX algorithm is able to extract a wide variety of parameters, but places significant emphasis on onset and offset detection, as it contributes to many temporal feature extractions, such as tone rate, articulation and onset velocity [5]. Hence, we chose tone rate to provide a rough tempo estimation, and fluctuations in the tone rate are used to identify temporal expressive intentions for our A-DAFx.

Timbral deviations also play an important role in enhancing the expressive value of a performance, and have a significant effect on the interpretation of music. The brightness attribute for example, has a direct correlation to the input force or energy of the sound. As a result, musicians are able to utilise this parameter for control and expression, which demonstrates a relation between acoustical timbre and expressive intention [7].

Loudness, another expressive perceptual attribute, which is related to the energy of a signal. Although amplitude variations do not influence expressivity to the same extent as temporal variations, they still have a significant effect, and as the energy of a signal can be intentionally controlled, it serves as a powerful tool for musical expression. Contrary to loudness, energy is not influenced by frequency, but this low level feature is able to provide sufficient information about the dynamics of the performance [6].

A guitar, as a string instrument produces nearly harmonic sounds, and its pitch can be determined as a result [1]. Harmonies and melodies carry important information on musical structure [3]. Yet musical structure is often established by a composer, therefore variations in pitch are not necessarily related to expressive performance. A focus on improvisational performance however, would allow the investigation of temporal changes in pitch as an expressive detail. The physical counterpart of pitch is the fundamental frequency, which can serve as an adequate indicator.

3. DESIGN

Prior work provided us with an overall set of requirements on developing an A-DAFx, and algorithms for extracting expressive acoustic features.

We found the field of A-DAFx to have an emphasis on implementation of algorithms and mapping strategies, while comparably few user-oriented studies have been done in the field, which highlights a potential area of focus. DAFx are commonly used by electric guitar players, making them an ideal main target group for the purpose of this project. This led to an investigation on how to extract information and expressive features from the electric guitar and how to map these features to the A-DAFx parameters. A set of design criteria were formed to frame and guide the project with a special emphasis on implementing A-DAFx that would be beneficial for electric guitar players.

Design Requirements:

- It should function in most Digital Audio Workstations (DAWs) as a VST format.
- The plugin should work in real-time without notable latency.
- The extracted features should be generalised and mapped to work with several different effects.

For the purpose of creating an A-DAFx, a digital implementation of the spatial 'echo' effect was chosen. To create a more interesting echo effect, a variety of modulation and timbral effects were implemented into the echo engine. These included highpass and lowpass filters, vibrato, reverse delay, and saturation (tube). The choice of effects was influenced by commercial delay plugins that utilise similar effects, for example: Strymon's *'Timeline'*, ChaseBliss Audio's *'Tonal recall'*, and Red Panda's *'Particle Granular Delay/Pitch Shifter'* [17].

During the design phase, the DAFx was tested and tweaked using MATLAB by MathWorks and its Audio System Toolbox Library. The white paper by Mathsworks' DeVane and Bunkhelia served as inspiration on how to work with the library [18].

4. IMPLEMENTATION

4.1 Real-Time DAFx

Since the adaptive effect should work in real-time, some modifications had to be made in order for the application to work. Real-time audio processing follows a frame-based pipeline where the signal is segmented into buffers for processing and output [19]. The frame size (also known as buffer size) and sampling rate is unknown, since these parameters can be changed by the user. To prevent crashing due to using an input size or sampling rate that is too large or small, the program needs to adjust itself and work with variable sizes. Furthermore, efficiency needs to be addressed since the user should not experience any notable latency during playing. If the effect and feature extraction algorithms require too much computational power, the input will output to the DAC too late, causing overrun and undesired output. Although it introduces latency, this problem can be fixed by using a larger frame size. Efficiency however was not a primary concern during the project implementation.

4.2 Delay Based Effects

4.2.1 Saturation

The saturation algorithm is based on Zölzer's implementation of tube distortion [4, pp. 122-123]. This algorithm replicates the nonlinearities introduced in valve amplifiers by asymmetrical clipping of the input audio signal x.

$$f(x) = \begin{cases} \frac{x-Q}{1-e^{-dist}(x-Q))} + \frac{Q}{1-e^{dist\cdot Q'}}, & Q \neq 0, x \neq Q, \\ \frac{1}{dist} + \frac{Q}{1-e^{dist\cdot Q'}} & x = Q \end{cases} \tag{1}$$

,

At low input levels there should be no distortion in the signal. Clipping and limiting should occur at large negative input values and be approximately linear for positive values [3]. The work point Q, controls how much nonlinear clipping is introduced in the signal. Large negative values for Q will provide a more linear output, and dist increases the amount of distortion in the signal.

4.2.2 Vibrato

The implementation for the vibrato was inspired by Zölzer's et al. vibrato algorithm [3]. This algorithm produces vibrato in an audio signal by creating pitch variations in a way that is similar to the Doppler effect. Varying the distance between the sound source and the listener is the same as varying the delay in the signal. If the variation in the delay is periodic, then the pitch variation is also periodic.

To implement this, a circular buffer is used with a read and write index. The write index is used for writing the audio input in the buffer. The read index reads the delayed audio input from the buffer in an oscillating manner creating periodic variations in the outputted delay.

4.2.3 Reverse Delay

The inspiration for this type of effect stemmed mainly from commercial pedals stated previously. The effect was achieved by saving the input in a delay line, reversing it and then reading from it. A variable was stored to keep track of the delay time in samples with the purpose of using and reversing the necessary buffer length. Pointer variables, a write index and a read index, were used to keep track of where the input was stored in the delay line, and what was required to be sent to the audio output.

4.2.4 Filters

One very common design for a low-pass filter, which can be easily implemented, is the biquad filter. This filter is a second-order linear IIR filter with two poles and two zeroes. One big advantage of using a second-order filter over a high-order IIR filter is greater stability [20]. The filter can be controlled with three user-defined parameters: the sampling frequency f_s, the corner frequency f_0, and the quality factor Q [21].

4.3 Expressive Feature Extraction

4.3.1 Pitch Detection:

The implemented pitch detection algorithm was inspired by the 'Harmonic Product Spectrum (HPS)' technique and is used for extracting the fundamental frequency for any given audio input signal in real-time. The main reason for choosing this method was due to its ability to run in real-time and perform well under a wide range of conditions [22]. An audio signal spectrum will usually consist of a series of peaks, corresponding to the fundamental frequency with harmonic components at integer multiples of the fundamental [23]. The HPS method utilises this by down-sampling the spectrum a number of times, multiplying the compressed versions together, and comparing it to the original spectrum. The resulting spectrum will display an alignment of the strongest harmonic peaks and consequently, the fundamental frequency. The HPS technique measures the maximum coincidence for harmonics for each spectral frame according to:

$$Y(\omega) = \prod_{r=1}^{R} |X(\omega r)| \tag{2}$$

Where R is the number of harmonics being considered, The resulting periodic correlation array is then searched for a maximum value of a range of possible fundamental frequencies to obtain the fundamental frequency estimate [22].

$$\widehat{Y} = arg\ maxY(\omega_i) \tag{3}$$

4.3.2 *Spectral Centroid:*

The spectral centroid provides a strong impression of the timbral feature; brightness. It is calculated by taking a spectrum's centre of gravity according to the following formula [24,25].

$$SC_{hz} = \frac{\sum_{k=1}^{N-1} k * X^d[k]}{\sum_{k=1}^{N-1} X^d[k]} \tag{4}$$

The calculation was implemented by applying a hamming window for each buffer, to attenuate any unwanted frequencies that result from taking the Fourier Transform of a segment of the audio signal. For music signals, the brightness generally increases with the presence of higher frequency components in the signal, which is often introduced through inharmonic sounds [25,26].

4.3.3 *Energy:*

Energy is a basic but powerful acoustic feature as it describes the signal energy [27]. Energy is calculated by squaring and summing up amplitudes in a windowed signal, after which the sum is normalised by the length of the window.

$$E_n = \frac{1}{N} \sum_{n=0}^{N-1} x^2(n) \tag{5}$$

The frame size of the input signal was used to decide the window length. The value generated by the energy was used to estimate the amount of force in the guitar strumming.

4.3.4 *Tone Rate:*

Tone rate was implemented to gain information on the performers speed of playing. It is roughly proportional to tempo, as it refers to the number of successive tones over a given period, but this range is not large enough to serve as a detailed indication of tempo. In order to gain a better resolution, we derive the average inter onset interval (IOI) to indicate speed at which the musician is playing. The onset marks a single point in time within the signal where the transient region or attack starts, highlighting a sudden burst of energy [28].

Onset detection based on spectral features include various robust methods, which generally require less preprocessing and potentially allow the bandwise analyses of polyphonic signals. The reduction function transforms the set of spectral windows into a so called novelty curve, which indicates the location of prominent changes, and therefore the possible onsets. The novelty curve is acquired from the rectified spectral difference at each window, treating the onset as a broadband event, capturing only positive changes in the magnitude spectrum [28,29].

$$SD = \sum_{k=0}^{K} |X[k]| - |X[k-1]|| \tag{6}$$

The localisation of an onset relies on two threshold parameters. First, an absolute threshold on the amount of positive change in the magnitude spectrum, to filter out the spurious peaks. The second being temporal threshold, which makes sure that only one onset is detected in a given range. In order to calculate the average inter-onset interval, the onset intervals are accumulated over time in a vector and divided by the total number of onsets in the same period.

4.4 Parameter Mapping

In order to demonstrate the capabilities of the effect we created a specific preset. The parameter mappings for the pre-

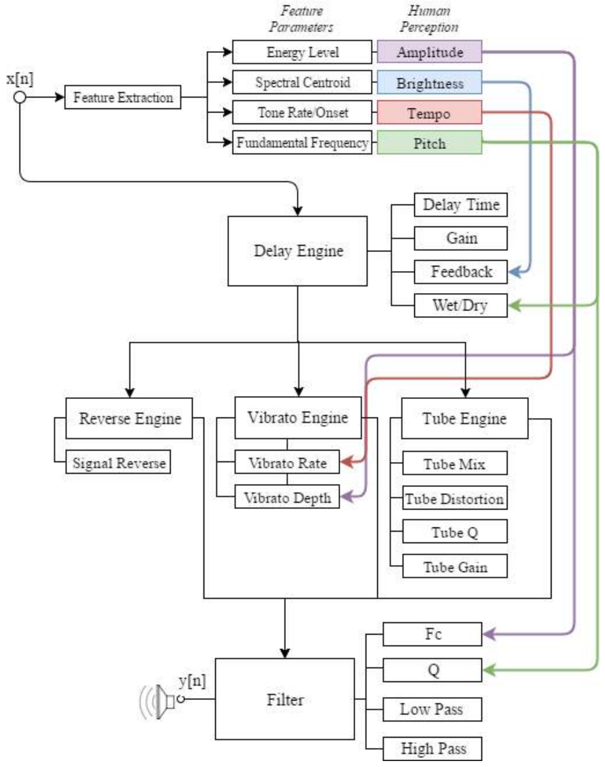


Figure 2. Effect engine and parameter mapping structure

set called 'Dreamy', were determined by an attempt to construct meaningful relationships between feature extractions and effect controls, and to also create an accurate representation of the preset name. The notion of dreamy music is often associated with rich ambient and reverberant sounds, which can be digitally replicated by utilising effects such as delay, reverb and vibrato. An attempt to achieve this type of 'sound identity' was made through the parameter relationships described. We found adaptive mappings such as 'Pitch Height → Delay Mix' to make sense musically, as overly delayed signals in low frequencies can create an undesired muddy low end. The mappings were decided from what we found aesthetically pleasing and fitting to the chosen notion of dreamy. Further, to inform the choice of adaptive mapping, the dreamy preset was subject to initial tests. The non-adaptive versions static parameters were centred in relation to the range of the adaptive mapping. Figure 2 displays the signal path of the whole effect, with the coloured lines outlining the parameter mappings for 'Dreamy'. The adaptive mappings for the 'Dreamy' preset are as follows:

- Energy Level → Vibrato Depth + Filter Fc
- Spectral Centroid → Delay Feedback
- Tone/Rate → Vibrato Rate
- Fundamental Frequency → Delay Mix + Filter Q

5. EVALUATION

Having designed and implemented an adaptive digital audio effect that uses expressive audio features as input, we now turn to evaluating its performance and use for a performer. In the following we will describe the evaluation which was done to aswer the research question: Are users able to distinguish between the adaptive and non-adaptive version, and do they understand what the difference is?

5.1 Participants

In order to take part in the evaluation, we required that the participant be able to perform basic guitar techniques. Additionally, we deemed knowledge about effects and various musical terms necessary for the participant to understand the questions asked during the experiment. Hence the target group was defined as intermediate guitar players well acquainted with audio effects.

The participants were recruited through purposive sampling [30]. The researchers chose associative participants, who fit the target group. A total of seven people participated in the evaluation. The participants were between 22 and 27 years old and the average guitar playing experience was 9.8 years.

5.2 Procedure

We chose a within-participant design where players compared two versions of our effect, one adaptive and one non-adaptive. In order not to affect the behaviour of the participants in a systematic way [31], the presentation order of the two versions was counterbalanced so that Participant 1 tested with A = non-adaptive, B = adaptive, while Participant 2 tested with A = adaptive, B = non-adaptive, etc.).

Initially a pilot test was performed, and the experimental procedure was adjusted accordingly. The final procedure was deducted as follows: Each participant was given a short scripted introduction. They were told they would be trying two versions of an effect, and were introduced to the plugin interface. Thereafter the participant would be asked to sign a terms of agreement, allowing for the experiment to be recorded. A small demographic questionnaire was also conducted to obtain information on age, experience, and preferred genre. This was done to make sure that participants fit the chosen target group.

The remaining test was structured as follows:

1. Basic tasks; 2. Short interview; 3. Expressive improvisational tasks; and 4. Follow-up interview.

In the first set of tasks, the participant was asked to perform a variety of basic playing styles on the guitar. The goal of these tasks was to make sure that the participant explored every aspect of the effect, and was able to gather some thoughts about the difference between the two versions. After performing the basic tasks, a short interview was conducted, where the participant was asked to explain the difference between the two versions. There were questions specifically asking about the effect of certain features (such as strum power or pitch), which helped clarify if the participant understood changes in the sound.

The next set of tasks focused on the participants own interpretation of expressiveness. They were asked to try version A and B when playing deadpan and expressively. The prediction was that the difference between A and B would be more noticeable when playing expressively. Deadpan was described as mechanical playing, adding no expressive variation to the playing. Expressively was defined as performing as though you were trying to convey some musical expression and emotion to an audience.

Finally, another short interview was conducted. The participant was asked to explain if the difference between version A and B was more noticeable when playing expressively. Afterwards the difference was revealed by the experimenter, and the participant was asked to guess which of the versions was the adaptive one. The complete test lasted between 25-35 minutes per participant.

5.3 Data Collection

1) Qualitative Interview: The main method for gathering data was through qualitative structured interviews: All of the questions required the participants to provide a specific answer (yes, no, A or B). This allowed for a structured comparison between all their answers. Additionally, the participants were encouraged to elaborate on each question, with the test experimenter taking notes of their answers throughout the interviews.

2) Observation: The tests were all video recorded for post-observation. The guitar was recorded with the effects applied. The recordings served several purposes:

1. Review the interviews, in case the test experimenter missed any important notes. 2. Check for user errors did they understand the tasks correctly? 3. Check for systematic errors glitches, and other types of unpredictable behaviour. 4. Triangulate with the data from the interviews.

Question asked	Choice Adapt.	Choice Non-Adap.	Unsure
Q1: Perceived noticeable difference?	Yes: **7**	No: 0	0
Q2: In what version did the energy level change the effect?	**6**	1	0
Q3: In what version did the tempo change the effect?	1	1	**5**
Q4: In what version did the pitch change the effect	**4**	1	2
Q5: Were you at any point confused?	Yes:3	No:**4**	0
Q6: In which version did you have more control?	**5**	2	0
Q7: More noticeable difference when playing expressively?	**6**	1	0
Q8: Which version made you motivated to play expressively?	3	3	1
Q9: Preferred version	**4**	3	0
Q10: Which was adaptive?	**5**	1	1

Table 1. Summary of the participant's choices to the simple questions during the evaluation. The questions have been condensed here, but can be found their actual format in the appendix. The data in 'bold' outlines the successful choices in the interview. Ad = the adaptive version. Non-Ad = the non-adaptive version. U = Unclear, Dont know or No perceived difference.

5.4 Analysis Methods

As a first step, participants' choices to the simple questions were summarised (See table 1), after which a more thorough analysis of the interview was performed using Meaning Condensation and Content Analysis. The full interview answers were reduced by performing meaning condensation, after which the content was analysed by searching for keywords, patterns, and outliers to gain a broad perspective of the answers. To strengthen the validity of the study, the data from the interviews was compared with the observation data. If the two sources are not congruent we considered the data to be invalid.

6. RESULTS

6.1 Difference Noticed Between Versions

1) General: All participants perceived a difference between the adaptive and the non-adaptive version. The non-adaptive version was described as a flat, dry and more simple compared to the adaptive. The adaptive version was described to have a thicker/fuller effect where there was more happening with the effect.
2) Effect Change According to Energy Level: A vast majority said that the power of their strum changed some parameter of the effect in adaptive version. Generally, some parameter of the effect changed according to the power of the strum. There were different interpretations about what happened, some of which were: sensitivity of an envelope filter, added harmonics, and presence.
3) Effect Change According to Tempo: There was a lot of confusion regarding the mapping of the tempo. Generally, the participants were unable to perceive a difference from their tempo fluctuations. The participants could not tell the difference between the adaptive version and the non-adaptive version.
4) Effect Change According to Pitch: There was some confusion regarding the mapping of the pitch. Most were able to hear a change of the effect in the adaptive version, however some could not perceive a difference between the two versions. A change in the adaptive version was identified, but the description of the effect was inconsistent between participants. They could hear that something happened but could not define exactly what it was.

6.2 Comparing 'Deadpan' to 'Expressive' Playing

6.2.1 Expressive Motivation:

The version that made the participants more motivated to play expressively appeared to be random, and the version that participants preferred in general was fairly random. There was no agreed preference. The non-adaptive version was deemed simpler and predictable, while the adaptive version was deemed playful, stronger, and interesting for experimenting with sounds.

6.2.2 More Noticeable Differences:

All, except for one participant, thought that the difference between the two versions was more noticeable when playing expressively compared to deadpan. The majority of participants guessed correctly when asked which the adaptive version was. Participants thought there was something adaptive happening in both versions when playing expressively, but found it more clear in the adaptive version. Comments include that the effect seemed more static in both versions when playing in a deadpan manner.

6.3 Usability

Some participants were confused about certain aspects of the effect, but most found it intuitive. Most participants felt that they had more control in the adaptive version, but it was difficult to determine exactly what happened with the sound. The adaptive version was deemed complex in the sense that there were more things going on, while the non-adaptive was deemed simpler in terms of control and understanding.

7. DISCUSSION

The results showed that the participants were able to perceive a notable difference between the two versions of the effect, and they were able to tell which one was adaptive. However, they had trouble understanding exactly what features actually affected the parameters of the effect. To gain an understanding of why some of these issues occurred, triangulation was performed.

7.1 General Comparison Between Versions

The participants described the non-adaptive version as flat, dry and more simple compared to the adaptive. The adaptive version was described to have a thicker/fuller effect where there was more happening with the effect. The influence of the energy was the most noticeable and a possible reason for this could be the direct relation between the power of the strum and the perceived changes in the vibrato depth and the cut off frequency. The influence of playing tempo and the pitch on the effect was less clear to the participants. The players could detect the influence of pitch, but were not able to identify what parameter it was controlling. The least successful mapping was the tempo and the vibrato rate, which the participants were perceiving the same in both cases. The noticed difference can also be biased by the imbalanced influence of different mappings, therefore the masking of one effect of another.

In the more free task involving expressive playing, players thought both the non-adaptive and the adaptive versions were adaptive. But the players described the non-adaptive version to be simpler and predictable, while the adaptive version was deemed playful, stronger, and interesting for experimenting with sounds.

7.2 Further Comments

The responses to the questions regarding general usability that participants answered at the end of the test can be helpful to set design requirements for further development. All the participants thought that an adaptive effect would be useful, but mostly for experimental purposes. They generally wished that they had more control, or at least knew what kind of parameters were active in the effect, which highlights a clear need for transparency in the further design of such tool. The participants especially liked the idea of being able to control the intensity of 'adaptive expression' applied to the effect. They also had some ideas for mapping the effect differently.

8. CONCLUSIONS

An adaptive digital audio effect was implemented to run real-time and was tested on intermediate guitar players. The effect was a multi-dimensional delay engine with added vibrato, reverse delay, saturation, and filters. Several sound features, pitch, brightness, tone rate and energy were extracted from the input signal and transformed the output signal by varying the effect parameters over time. The design aimed to exploit the musical intention of the players in order to manipulate effect parameters. Through testing it was found that the participant generally did not find that the adaptive version made them play more expressively than the non-adaptive, and that users did not prefer the adaptive effect. However, when using the adaptive version, the participants felt they had more control and noticed a bigger difference in the sound produced by the effect when playing expressively compared to deadpan. It was also found that the mapping of tempo and pitch were not apparent to the users, whereas the energy mapping was easily recognised. To investigate A-DAFx further, it would be worthwhile to test the effect in various environments (such as in a recording or live setting). In conjunction with this, it may be beneficial to apply a longitudinal testing approach so that participants could integrate the effect in their own creative and compositional work process.

Acknowledgments

The authors greatfully thank the guitarists involved in the evaluation.
Authors Holfelt, Csapo, Andersson, Zabetian, and Castanieto were responsible for the main work which was performed as a student project during the first semester of Sound and Music Computing, Aalborg University Copenhagen. Authors Erkut and Dahl supervised the project and participated in the writing together with author Overholt.

9. REFERENCES

[1] D. M. Campbell, C. A. Greated, A. Myers, and M. Campbell, *Musical Instruments: History, Technology, and Performance of Instruments of Western Music*. Oxford: Oxford University Press, 2004.

[2] J. Bruck, A. Grundy, and I. Joel, "An Audio Timeline," 2016, accessed: Dec. 15, 2016. [Online]. Available: http://www.aes.org/aeshc/docs/audio.history.timeline.html

[3] U. Zölzer, *DAFX: Digital Audio Effects*, 2nd ed. Oxford, England: Wiley, John & Sons, 2011.

[4] V. Verfaille and D. Arfib, "A-DAFX: Adaptive Digital Audio Effects," *Digital Audio Effects (DAFX-01)*, 2001.

[5] A. Friberg, E. Schoonderwaldt, and P. Juslin, "Cuex: An Algorithm for Automatic Extraction of Expressive Tone Parameters in Music Performance from Acoustic Signals," *Acta Acustica United With Acustica*, vol. 93, no. 3, 2007.

[6] A. Bhatara, A. K. Tirovolas, L. M. Duan, B. Levy, and D. J. Levitin, "Perception of Emotional Expression in Musical Performance," *Journal of Experimental Psychology: Human Perception and Performance*, vol. 37, no. 3, 2011.

[7] M. Barthet, R. Kronland-Martinet, and S. Ystad, "Improving Musical Expressiveness by Time-Varying Brightness Shaping," *Computer Music Modelling and Retrieval*, vol. 4969, 2008.

[8] R. A. Kendall and E. C. Carterette, "The Communication of Musical Expression," *Music Perception: An Interdisciplinary Journal*, vol. 8, no. 2, 1990.

[9] L. Mion and G. D'Incà, "Analysis of Expression in Simple Musical Gestures to Enhance Audio in Interfaces," *Virtual Reality*, vol. 10, no. 1, 2006.

[10] C. Oshima, K. Nishimoto, Y. Miyagawa, and T. Shirosaki, "A Concept to Facilitate Musical Expression," *Proceedings of the 4th conference on Creativity & cognition*, 2002.

[11] U. Z. V. Verfaille and D. Arfib, "Adaptive Digital Audio Effects (a-DAFX): A New Class of Sound Transformations," *IEEE Transactions on Audio, Speech and Language Processing*, no. 5, 2006.

[12] V. Verfaille, M. M. Wanderley, and P. Depalle, "Mapping Strategies for Gestural and Adaptive Control of Digital Audio Effects," *Journal of New Music Research*, vol. 35, no. 1, 2006.

[13] B. Corporation, "Te-2 — Tera Echo," 2016, accessed: Dec. 13, 2016. [Online]. Available: https://www.boss.info/us/products/te-2/

[14] ——, "Da-2 — Adaptive Distortion," 2016, accessed: Dec. 13, 2016. [Online]. Available: https://www.boss.info/us/products/da-2/features/

[15] ——, "Mo-2 — Multi Overtone," 2016, accessed: Dec. 13, 2016. [Online]. Available: https://www.boss.info/us/products/mo-2/

[16] M. Müller, *Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications: 2015*. Switzerland: Springer International Publishing AG, 2015.

[17] G. Tanaka, "Reviews of the Best Guitar Pedals & Gear," 2013, accessed: Dec. 12, 2016. [Online]. Available: http://www.bestguitareffects.com/top-best-delay-guitar-effects-pedals-buyers-guide/

[18] C. DeVane and G. Bunkheila, "Automatically Generating VST Plugins from MATLAB Code," 2016, accessed: Dec. 12, 2016. [Online]. Available: http://www.aes.org/e-lib/browse.cfm?elib=18142

[19] MathWorks, "Audio I/O: Buffering, Latency, and Throughput," 2016, accessed: Dec. 12, 2016. [Online]. Available: https://se.mathworks.com/help/audio/gs/audio-io-buffering-latency-and-throughput.html

[20] T. Instruments, "Configure the Coefficients for Digital Biquad Filters in TLV320AIC3xxx family," 2010, accessed: Dec. 12, 2016. [Online]. Available: http://www.ti.com/lit/an/slaa447/slaa447.pdf

[21] R. Bristow-Johnson, "Cookbook Formulae for Audio EQ Biquad Filter Coefficients," 2016, accessed: Dec. 12, 2016. [Online]. Available: http://www.musicdsp.org/files/Audio-EQ-Cookbook.txt

[22] T. Smyth, "Music 270a: Signal Analysis," 2015, accessed: Dec. 12, 2016. [Online]. Available: http://musicweb.ucsd.edu/~trsmyth/analysis/analysis.pdf

[23] G. Middleton, "OpenStax CNX," 2003, accessed: Dec. 12, 2016. [Online]. Available: http://cnx.org/contents/i5AAkZCP@2/Pitch-Detection-Algorithms

[24] T. H. Park, *Introduction to Digital Signal Processing*, 1st ed. Singapore: World Scientific Publishing Co., 2010.

[25] E. Schubert, "Does Timbral Brightness Scale with Frequency and Spectral Centroid?" *Acta Acustica United With Acustica*, vol. 92, 2006.

[26] H. Järveläinen and M. Karjalainen, "Importance of Inharmonicity in the Acoustic Guitar," *Forum Acusticum*, vol. 2005, 2005.

[27] F. Eyben, *Real-time Speech and Music Classification by Large Audio Feature Space Extraction*. Springer, 2016.

[28] J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. B. Sandler, "A Tutorial on Onset Detection in Music Signals," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 5, 2005.

[29] M. Müller, D. Ellis, A. Klapuri, and G. Richard, "Signal Processing for Music Analysis," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 6, 2011.

[30] T. Bjørner, *Qualitative Methods for Consumer Research: The Value of the Qualitative Approach in Theory & Practice*. Denmark: Reitzel, Hans, 2015.

[31] A. Field and G. J. Hole, *How to Design and Report Experiments*. London: SAGE Publications, 2003.

Virtual Analog Model of the Lockhart Wavefolder

Fabián Esqueda, Henri Pöntynen
Dept. of Signal Processing and Acoustics
Aalto University
Espoo, Finland
fabian.esqueda@aalto.fi
henri.pontynen@aalto.fi

Julian D. Parker
Native Instruments GmbH
Berlin, Germany
julian.parker@native-instruments.de

Stefan Bilbao
Acoustics and Audio Group
University of Edinburgh
Edinburgh, UK
s.bilbao@ed.ac.uk

ABSTRACT

This work presents a novel virtual analog model of the Lockhart wavefolder. Wavefolding modules are among the fundamental elements of 'West Coast' style analog synthesis. These circuits produce harmonically-rich waveforms when driven by simple periodic signals such as sinewaves. Since wavefolding introduces high levels of harmonic distortion, we pay special attention to suppressing aliasing without resorting to high oversampling factors. Results obtained are validated against SPICE simulations of the original circuit. The proposed model preserves the nonlinear behavior of the circuit without perceivable aliasing. Finally, we propose a practical implementation of the wavefolder using multiple cascaded units.

1. INTRODUCTION

In 1965, Bob Moog (1934–2005) presented his seminal work on the design of a voltage-controlled filter (VCF) at the 17th Annual AES Meeting [1]. Moog's design became a key element of the celebrated Moog sound and of electronic music in general. His work paved the way for the development of a synthesis style known as ”East Coast” synthesis, named after Moog's New York origins. Two years prior, in 1963, the San Francisco Tape Music Center, along with composers Morton Subotnick and Ramon Sender, commissioned a voltage-controlled instrument from the Berkeley-based Don Buchla (1937–2016). This led to the development of Buchla's first synthesizer, the Buchla 100, and the birth of ”West Coast” synthesis [2].

Although contemporaries, the synthesis paradigms of Moog and Buchla had very little in common. In East Coast synthesis, sounds are sculpted by filtering harmonically-rich waveforms, such as sawtooth or square waves, with a resonant filter. This approach is known in the literature as subtractive synthesis. In contrast, West Coast synthesis eschews traditional filters and instead manipulates harmonic content, or timbre, at oscillator level using a variety of techniques such as waveshaping and frequency modulation. The resulting waveforms are then processed with a lowpass gate (LPG), a filter/amplifier circuit that uses photoresistive opto-isolators, or vactrols, in its control path [3]. To set them apart from their East Coast counterparts, West Coast oscillators are called ”complex oscillators”.

One of Buchla's early waveform generators featured a wavefolding circuit designed to control timbre. Wavefolding is a type of nonlinear waveshaping where portions of a waveform are inverted or ”folded back” on itself. When driven by a signal with low harmonic content, e.g. a sine or triangular oscillator, wavefolders can generate harmonically-rich waveforms with distinctive tonal qualities. This work presents a novel virtual analog (VA) model of the Lockhart wavefolder, a West Coast-style circuit proposed by Ken Stone as part of his CGS synthesizer and available as a DIY project on his personal website [4].

Recent years have seen an increase in the number of manufacturers embracing West Coast synthesis and releasing their own takes on classic Buchla and Serge (another famed West Coast designer of the 1970s) modules. Modern synthesizer makers, such as Make Noise, Intellijel and Doepfer, all feature complex oscillators and LPGs in their product lines. This growing interest in modular synthesizers, which are generally exclusively expensive, serves as the principal motivation behind the development of VA models of these circuits. VA synthesizers are generally affordable and are exempt from the inherent limitations of analog circuits, e.g. faults caused by aging components.

An essential requirement in VA modeling is to preserve the ”analog warmth” of the original circuit [5,6]. This perceptual attribute is associated with the nonlinear behavior inherent to semiconductor devices and vacuum tubes, and can be modeled via large-signal circuit analysis. This approach has been researched extensively in the context of VCFs [7–11] and effects processing [12–19]. The use of nonlinear waveshaping in digital sound synthesis is also well documented [20–23].

A particular challenge in VA models of nonlinear devices is aliasing suppression. High oversampling factors are usually necessary to prevent harmonics introduced by nonlinear processing from reflecting to the baseband as aliases. Aliasing reduction has been widely studied in the context of waveform synthesis [24–30]. Moreover, recent work has extended the use of a subset of these techniques to special processing cases such as signal rectification and hard clipping [31,32]. The proposed Lockhart VA model incorporates the antiderivative antialiasing method proposed by

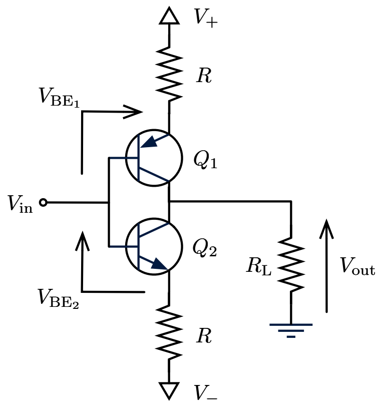


Figure 1. Schematic of the Lockhart wavefolder circuit.

Parker et al. [33, 34].

This paper is organized as follows. Sections 2 and 3 focus on the analysis of the Lockhart circuit and derivation of an explicit model. Section 4 deals with the digital implementation of the model. Section 5 discusses synthesis topologies based around the proposed wavefolder. Finally, Section 6 provides concluding remarks and thoughts for further work.

2. CIRCUIT ANALYSIS

Figure 1 shows a simplified schematic of the Lockhart wavefolder adapted from Ken Stone's design [4]. This circuit was originally designed to function as a frequency tripler by R. Lockhart Jr. [4, 35], but was later repurposed to perform wavefolding in analog synthesizers. The core of the circuit consists of an NPN and a PNP bipolar junction transistor tied at their base and collector terminals. In order to carry out the large-signal analysis of the circuit, we replace Q_1 and Q_2 with their corresponding Ebers-Moll injection models [10] as shown in Figure 2. Nested subscripts are used to distinguish between the currents and voltages in Q_1 from those in Q_2. For instance, I_{CD_1} is the current through the collector diode in Q_1.

We begin our circuit analysis by assuming that the supply voltages $V_\pm$ will always be significantly larger than the voltage at the input, i.e. $V_- < V_{\mathrm{in}} < V_+$. This assumption is valid for standard synthesizer voltage levels ($V_\pm = \pm 15$V, $V_{\mathrm{in}} \in [-5, 5]$V), and implies that the base-emitter junctions of Q_1 and Q_2 will be forward-biased with approximately constant voltage drops for all expected input voltages. Applying Kirchhoff's voltage law around both input–emitter loops (cf. Fig. 1) results in

$$V_{\mathrm{in}} = V_+ - RI_{\mathrm{E}_1} - V_{\mathrm{BE}_1}, \tag{1}$$
$$V_{\mathrm{in}} = V_{\mathrm{BE}_2} + RI_{\mathrm{E}_2} + V_-, \tag{2}$$

where I_{E_1} and I_{E_2} are the emitter currents, and V_{BE_1} and V_{BE_2} are the voltage drops across the base-emitter junctions of Q_1 and Q_2, respectively. Solving (1) and (2) for the emitter currents then yields:

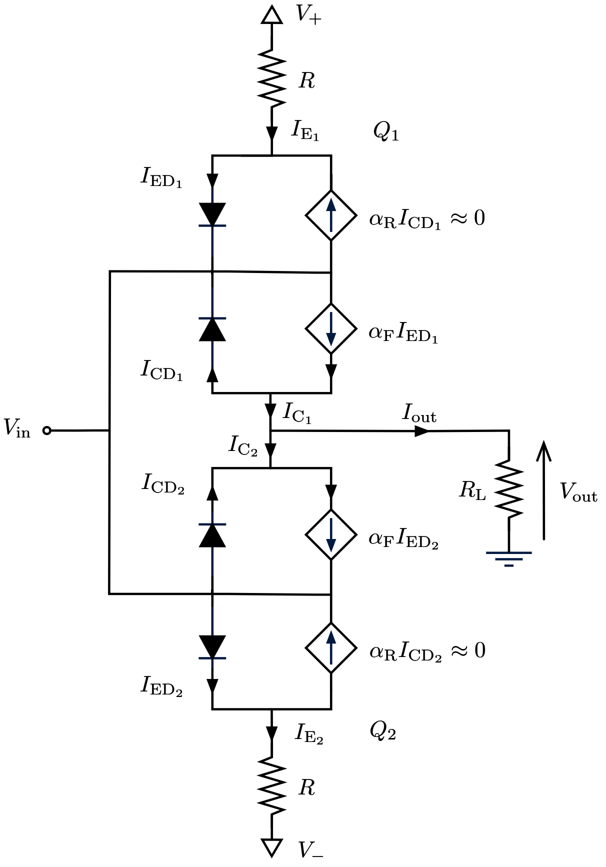


Figure 2. Ebers-Moll equivalent model of the Lockhart wavefolder circuit.

$$I_{\mathrm{E}_1} = \frac{V_+ - V_{\mathrm{BE}_1} - V_{\mathrm{in}}}{R}, \tag{3}$$
$$I_{\mathrm{E}_2} = \frac{V_{\mathrm{in}} - V_{\mathrm{BE}_2} - V_-}{R}. \tag{4}$$

Next, we apply Kirchhoff's current law at the collector nodes:

$$I_{\mathrm{out}} = I_{\mathrm{C}_1} - I_{\mathrm{C}_2}, \tag{5}$$
$$I_{\mathrm{C}_1} = \alpha_{\mathrm{F}} I_{\mathrm{ED}_1} - I_{\mathrm{CD}_1}, \tag{6}$$
$$I_{\mathrm{C}_2} = \alpha_{\mathrm{F}} I_{\mathrm{ED}_2} - I_{\mathrm{CD}_2}. \tag{7}$$

Assuming that the contribution of reverse currents $\alpha_R I_{\mathrm{CD}_1}$ and $\alpha_R I_{\mathrm{CD}_2}$ to the total emitter currents I_{ED_1} and I_{ED_2} is negligible, we can establish that

$$I_{\mathrm{ED}_1} \approx I_{\mathrm{E}_1} \qquad \text{and} \qquad I_{\mathrm{ED}_2} \approx I_{\mathrm{E}_2}.$$

Substituting these values in (6) and (7), and setting the value $\alpha_{\mathrm{F}} = 1$ gives a new expression for the output current:

$$I_{\mathrm{out}} = I_{\mathrm{E}_1} - I_{\mathrm{E}_2} - I_{\mathrm{CD}_1} + I_{\mathrm{CD}_2}. \tag{8}$$

Combining (3) and (4) we can derive an expression for the difference between emitter currents. Since the voltage drops V_{BE_1} & V_{BE_2} across the base-emitter junctions are approximately equal (i.e. $V_{\mathrm{BE}_1} \approx V_{\mathrm{BE}_2}$) for the expected range of V_{in}, their contribution to the difference of emitter currents vanishes. Therefore,

$$I_{\mathrm{E}_1} - I_{\mathrm{E}_2} = -\frac{2V_{\mathrm{in}}}{R}. \tag{9}$$

Substituting (9) into (8) yields an expression for the total output current I_{out} in terms of the input voltage and the currents through the collector diodes:

$$I_{\mathrm{out}} = -\frac{2V_{\mathrm{in}}}{R} - I_{\mathrm{CD}_1} + I_{\mathrm{CD}_2}. \tag{10}$$

Now, the I-V relation of a diode can be modeled using Shockley's ideal diode equation, defined as

$$I_{\mathrm{D}} = I_{\mathrm{S}}\left(e^{\frac{V_{\mathrm{D}}}{\eta V_{\mathrm{T}}}} - 1\right), \tag{11}$$

where I_{D} is the diode current, I_{S} is the reverse bias saturation current, V_{T} is the thermal voltage and η is the ideality factor of the diode. Applying Shockley's diode equation to the collector diodes using ideality factor $\eta = 1$, and substituting into (10) gives us

$$I_{\mathrm{out}} = -\frac{2V_{\mathrm{in}}}{R} - I_{\mathrm{S}}\left(e^{\frac{V_{\mathrm{CD}_1}}{V_{\mathrm{T}}}} - e^{\frac{V_{\mathrm{CD}_2}}{V_{\mathrm{T}}}}\right). \tag{12}$$

Next, we use Kirchhoff's voltage law to define expressions for V_{CD_1} and V_{CD_2} in terms of V_{in} and V_{out}

$$\begin{aligned} V_{\mathrm{CD}_1} &= V_{\mathrm{out}} - V_{\mathrm{in}}, & (13)\\ V_{\mathrm{CD}_2} &= V_{\mathrm{in}} - V_{\mathrm{out}}, & (14) \end{aligned}$$

and replace these values in (12). This gives us

$$I_{\mathrm{out}} = -\frac{2V_{\mathrm{in}}}{R} - I_{\mathrm{S}}\left(e^{\frac{V_{\mathrm{out}}-V_{\mathrm{in}}}{V_{\mathrm{T}}}} - e^{\frac{V_{\mathrm{in}}-V_{\mathrm{out}}}{V_{\mathrm{T}}}}\right). \tag{15}$$

As a final step, we multiply both sides of (15) by the load resistance R_{L} and remove the exponential functions to produce an expression for the output voltage of the Lockhart wavefolder:

$$V_{\mathrm{out}} = -\frac{2R_{\mathrm{L}}V_{\mathrm{in}}}{R} - 2R_{\mathrm{L}}I_{\mathrm{S}}\sinh\left(\frac{V_{\mathrm{out}} - V_{\mathrm{in}}}{V_{\mathrm{T}}}\right). \tag{16}$$

3. EXPLICIT FORMULATION

Equation (16) describes a nonlinear implicit relationship between the input and output voltages. Its solution can be approximated using numerical methods such as Halley's method or Newton-Raphson. These methods have previously been used in the context of VA modeling, e.g. in [10, 13, 16].

In this work, we propose an explicit formulation for the output voltage of the Lockhart wavefolder derived using the Lambert-W function. The Lambert-W function [1] $W(x)$ is defined as the inverse function of $x = ye^y$, i.e. $y = W(x)$. Recent research has demonstrated its suitability for VA applications. Parker and D'Angelo used it to model the control circuit of the Buchla LPG [3]. Several authors have used it to solve the implicit voltage relationships of diode pairs [3,36,37]. As described in [36], $W(x)$ can be used to solve problems of the form

$$(A + Bx)e^{Cx} = D, \tag{17}$$

which have the solution

$$x = \frac{1}{C}W\left(\frac{CD}{B}e^{AC/B}\right) - \frac{A}{B}. \tag{18}$$

Since the collector diodes in the model are antiparallel, only one of them can conduct at a time [38]. Going back to (10), when $V_{\mathrm{in}} \geq 0$ virtually no current flows through the collector diode of Q_1 (i.e. $I_{\mathrm{CD}_1} \approx 0$). The same can be said for I_{CD_2} when $V_{\mathrm{in}} < 0$. By combining these new assumptions with (12)–(14), we can derive a piecewise expression for the Lockhart wavefolder:

$$V_{\mathrm{out}} = -\frac{2R_{\mathrm{L}}}{R}V_{\mathrm{in}} + \lambda R_{\mathrm{L}}I_{\mathrm{S}}\exp\left(\frac{\lambda(V_{\mathrm{in}} - V_{\mathrm{out}})}{V_{\mathrm{T}}}\right), \tag{19}$$

where $\lambda = \mathrm{sgn}(V_{\mathrm{in}})$ and sgn() is the *signum* function.

This expression is still implicit; however, it can be rearranged in the form described by (17), which gives us:

$$\left(\frac{2R_{\mathrm{L}}}{R}V_{\mathrm{in}} + V_{\mathrm{out}}\right)\exp\left(\frac{\lambda V_{\mathrm{out}}}{V_{\mathrm{T}}}\right) = \lambda R_{\mathrm{L}}I_{\mathrm{s}}\exp\left(\frac{\lambda V_{\mathrm{in}}}{V_{\mathrm{T}}}\right) \tag{20}$$

Solving for V_{out} as defined in (18) yields an explicit model for the Lockhart wavefolder:

$$V_{\mathrm{out}} = \lambda V_{\mathrm{T}}W\left(\Delta\exp\left(\lambda\beta V_{\mathrm{in}}\right)\right) - \alpha V_{\mathrm{in}}, \tag{21}$$

where

$$\alpha = \frac{2R_{\mathrm{L}}}{R}, \quad \beta = \frac{R + 2R_{\mathrm{L}}}{V_{\mathrm{T}}R} \quad \text{and} \quad \Delta = \frac{R_{\mathrm{L}}I_{\mathrm{S}}}{V_{\mathrm{T}}}. \tag{22}$$

An important detail to point out is that the output of the Lockhart wavefolder is out of phase with the input signal by 180°. This can be compensated with a simple inverting stage.

4. DIGITAL IMPLEMENTATION

The voltages and currents inside an electronic circuit are time-dependent. Therefore, the Lockhart wavefolder can be described as a nonlinear memoryless system of the form

$$V_{\mathrm{out}}(t) = f(V_{\mathrm{in}}(t)), \tag{23}$$

where f is the nonlinear function (21) and t is time. Equation (24) can then be discretized directly as

$$V_{\mathrm{out}}[n] = f(V_{\mathrm{in}}[n]), \tag{24}$$

where n is the discrete-time sample index.

[1] Strictly speaking, $W(x)$ is multivalued. This work only utilizes the upper branch, often known as $W_0(x)$ in the literature.

4.1 Aliasing Considerations

The highly nonlinear behavior of (21) poses a challenge for its accurate discrete-time implementation. As an arbitrary input waveform is folded, new harmonics will be introduced. Those harmonics whose frequency exceeds half the sampling rate, or the Nyquist limit, will be reflected into the baseband and will cause unpleasant artifacts. Oversampling by high factors is commonly employed to mitigate this problem but this approach increases the computational requirements of the system by introducing redundant operations.

In this work, we propose using the first-order antiderivative method presented in [33] and [34]. This method is derived from the analytical convolution of a continuous-time representation of the processed signal with a first-order lowpass kernel. The antialiased form for the Lockhart wavefolder is given by

$$V_{\text{out}}[n] = \frac{F(V_{\text{in}}[n]) - F(V_{\text{in}}[n-1])}{V_{\text{in}}[n] - V_{\text{in}}[n-1]}, \tag{25}$$

where $F()$ is the antiderivative of $f()$, the wavefolder function (21). This antiderivative is defined as

$$F(V_{\text{in}}) = \frac{V_{\text{T}}}{2\beta}\left(1 + W\left(\Delta \exp\left(\lambda\beta V_{\text{in}}\right)\right)\right)^2 - \frac{\alpha}{2}V_{\text{in}}^2. \tag{26}$$

Since the antiderivative of W is defined in terms of W itself, this form provides a cheap alternative to oversampling. This means the value W, which constitutes the most computationally expensive part of both (21) and (26), only has to be computed once for each output sample.

Now, when $V_{\text{in}}[n] \approx V_{\text{in}}[n-1]$ (25) becomes ill-conditioned. This should be avoided by defining the special case

$$V_{\text{out}}[n] = f\left(\frac{V_{\text{in}}[n] + V_{\text{in}}[n-1]}{2}\right), \tag{27}$$

when $|V_{\text{in}}[n] - V_{\text{in}}[n-1]|$ is smaller than a predetermined threshold, e.g. 10^{-6}. This special case simply bypasses the antialiased form while compensating for the half-sample delay introduced by the method.

4.2 Computing the Lambert-W Function

Several options exist to compute the value of $W(x)$. In fact, scripting languages such as MATLAB usually contain their own native implementations of the function. Paiva et al. [38] proposed the use of a simplified iterative method which relied on a table-read for its initial guess. In the interest of avoiding lookup tables, we propose approximating the value of the Lambert-W function directly using Halley's method, as suggested in [39]. To compute w_m, an approximation to $W(x)$, we iterate over

$$w_{m+1} = w_m - \frac{p_m}{p_m s_m - r_m}, \tag{28}$$

where

$$\begin{aligned} p_m &= w_m e^{w_m} - x, \\ r_m &= (w_m + 1)e^{w_m}, \\ s_m &= \frac{w_m + 2}{2(w_m + 1)}, \end{aligned}$$

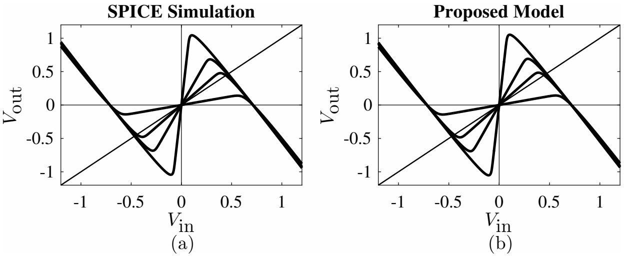


Figure 3. Input–output relationship of the Lockhart wavefolder circuit (inverted) measured using (a) SPICE and (b) the proposed digital model, for load resistor values $R_L = 1\text{k}$, 5k, 10k and 50kΩ (in order of increasing steepness).

and $m = 0, 1, 2, ..., M-1$. M is then the number of iterations required for p_m to approximate zero. The efficiency of the method will then depend on the choice of the initial guess w_0. An optimized MATLAB implementation of this method can be found in [40].

5. RESULTS

To validate the proposed model, the circuit was simulated using SPICE. Figure 3(a) shows the input–output relationship of the system measured with SPICE for values of V_{in} between –1.2 and 1.2 V and different load resistance values. Figure 3(b) shows the input–output relation of the proposed model (21) simulated using MATLAB. The polarity of the transfer function was inverted to compensate for the introduced phase shift. An ad hoc scaling factor of 2 was used on both measurements to compensate for the energy loss at the fundamental frequency. The results produced by the proposed model fit those produced by SPICE closely. As the value of the load resistor R_{L} is raised, the steepness of the wavefolder function increases. This increase in steepness translates into more abrasive tones at the output.

Figures 4(a) and (b) show the output of the wavefolder model when driven by a 500-Hz sinewave with a peak amplitude of 1 V, and $R_{\text{L}} = 10\text{k}$ and $50\,\text{k}\Omega$, respectively. Both curves are plotted on top of their equivalent SPICE simulations, showing a good match. The original input signal has been included to help illustrate the folding operation performed by the system.

Figure 5(b) shows the magnitude spectrum of a 1-V sinewave with fundamental frequency $f_0 = 2145\,\text{Hz}$ processed by the trivial (i.e. non-antialiased) wavefolder model (21). A sample rate $F_s = 88.2\,\text{kHz}$ (i.e. twice the standard 44.1 kHz audio rate), was used in this and the rest of the examples presented in this study. This plot shows the high levels of aliasing distortion introduced by the wavefolding process, even when oversampling by factor 2 is used. Figure 5(d) shows the magnitude spectrum of the same waveform processed using the antialiased form (25). As expected, the level of aliasing has been significantly reduced, with very few aliases left above the $-80\,\text{dB}$ mark. As illustrated by the left-hand side of Fig. 5, the antialiased form preserves the time-domain behavior of the system.

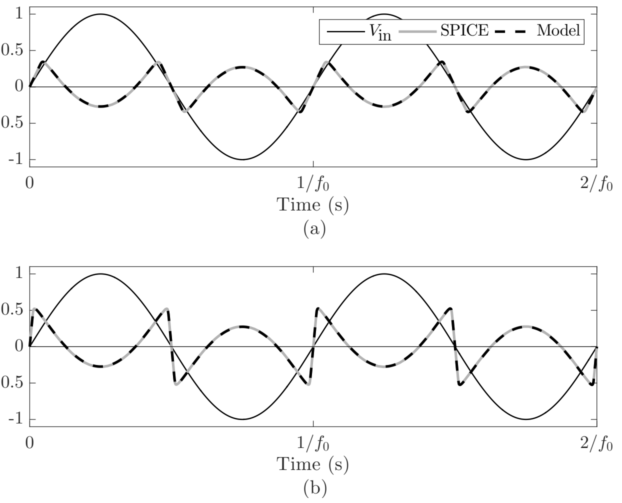


Figure 4. Time-domain view of the circuit's SPICE simulation vs the proposed model for a 500-Hz sinusoidal input with values of R_{L} = (a) $10\,\text{k}\Omega$ and (b) $50\,\text{k}\Omega$.

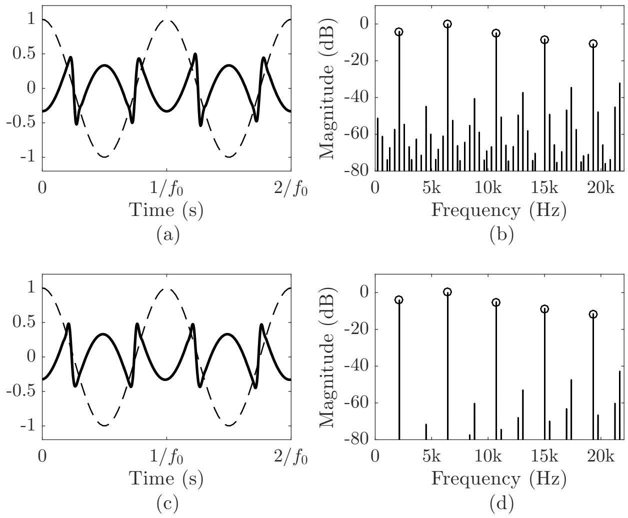


Figure 5. Waveform and magnitude spectrum of a 2145-Hz unit amplitude sinewave (a)–(b) processed trivially and (c)–(d) with the first-order antialiased form. Circles indicate non-aliased components. Parameter $R_{\text{L}} = 50\,\text{k}\Omega$.

The proposed antialiased form is particularly effective at reducing aliased components at low frequencies, especially below the fundamental. This behavior is illustrated in Fig. 6 which shows the logarithmic magnitude spectrum of a 4186-Hz sinewave (MIDI note C8 and highest fundamental frequency of a piano) processed both trivially and with the antialiasing form. The signal depicted by Fig. 6(a) suffers from a false perceived fundamental frequency at roughly 300 Hz. For the case of the antialiased signal in Fig. 6(b) this issue has been ameliorated. This low-frequency behavior is advantageous in our case because at low frequencies the audibility of aliasing distortion is only limited by the hearing threshold [41].

The performance of the antialiased model form was further evaluated by computing the A-weighted noise-to-mask ratio (ANMR) for a range of folded sinusoidal inputs. The ANMR has been previously suggested as a perceptually-informed measure to evaluate the audibility of aliasing distortion [28,41]. The algorithm computes the power ratio between harmonics and aliasing components, but takes into account the masking effects of the former. For instance, aliases clustered around harmonics will be rendered inaudible by the auditorty masking effects of such harmonics. This phenomenon is particularly common at high frequencies. An A-weighting filter is applied to all signals prior to evaluation to account for the frequency-dependent sensitivity of hearing for low-level sounds [41]. Signals with an ANMR value below $-10\,\text{dB}$ are considered to be free from audible aliasing.

Figure 7 compares the measured ANMRs of a set of unit-amplitude sinewaves with fundamental frequencies between 1–5 kHz processed by the wavefolder model using the antialasing form and different oversampling factors. The reference signals required to compute the ANMR values were generated using additive synthesis as detailed in [28]. All signals were downsampled back to audio rate (i.e. 44.1 kHz) prior to evaluation. This plot demonstrates that the performance of the proposed method, when combined with two-times oversampling, is on par with oversampling by factor 8. For all fundamental frequencies below approx. 4.2 kHz the ANMR lies below the $-10\,\text{dB}$ line. In terms of computational costs, the antialiasing method is approx. four times cheaper than trivial oversampling by 8 when measured under similar circumstances.

6. PRACTICAL SYNTHESIS USAGE

In practical sound synthesis applications an individual wavefolder is rarely used as the timbral variety it can produce is quite limited. Most designs, for example the Intellijel μFold, employ a number of wavefolding stages in series with intermediate gain elements used to space the folds. The number of stages varies between designs, and is in fact user variable in some cases. Typically somewhere between two and six folders are used.

Timbral control is then achieved using two parameters. The first is the gain at the input of the wavefolder. This parameter allows the overall brightness of the sound to be varied, and can be used to provide articulation to a sound similarly to a filter in subtractive synthesis or modulation index in FM synthesis. The second parameter is provided by adding a DC offset to the input of the wavefolder. This

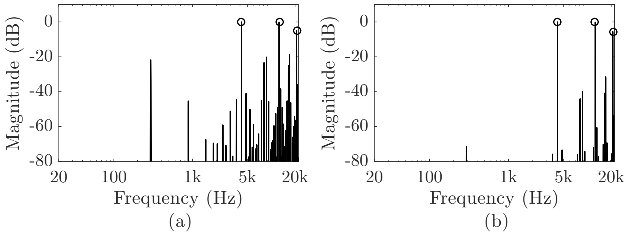


Figure 6. Log scale magnitude spectrum of a 4186-Hz sinewave (a) folded trivially and (b) with antialiasing. Parameter $R_{\text{L}} = 50\,\text{k}\Omega$.

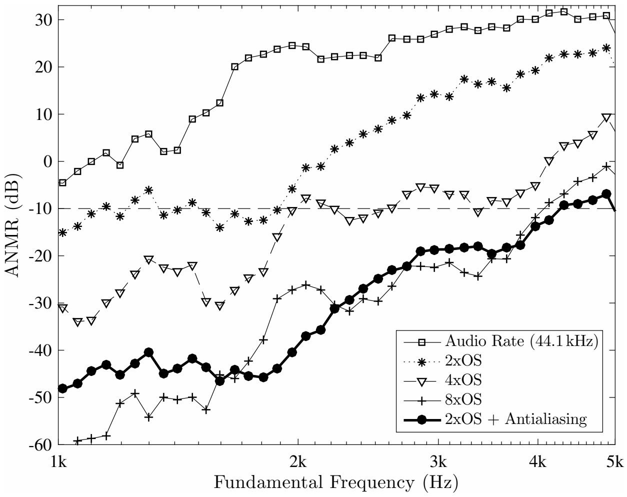


Figure 7. Measured ANMR of a range of sinusoidal waveforms processed using the Lockhart wavefolder model ($R_\mathrm{L} = 50\,\mathrm{k\Omega}$) under five different scenarios: direct audio rate, oversampling by factors 2, 3 and 8, and oversampling by 2 combined with the antialiasing form. Values below the –10 dB threshold indicate lack of perceivable aliasing.

breaks the symmetry of the folding and varies the relative amplitudes of the harmonics, without strongly effecting the overall brightness of the sound. This parameter can be modulated, e.g. with a low-frequency-oscillator (LFO), to provide an effect reminiscent of pulse-width modulation (PWM).

In order to build a well-behaved cascade of wavefolders, we need to make sure the individual folders satisfy two criteria. Firstly, the individual folders must provide approximately unity gain when $V_\mathrm{in} \approx 0$, and approximately negative unit gain beyond the folding point, when $|V_\mathrm{in}| >> 0$. Secondly, each stage should start folding at the same point with respect to its individual input. We can achieve this with the model described above by appropriate setting of R_L and the addition of static pre- and post-gain stages.

An appropriate R_L can be determined empirically. The pre- and post- gain can be determined by measuring the value of V_out at exactly the folding point. The pre-gain is taken to be approximately this value, and the post-gain is taken to be its inverse. In this case, $R_\mathrm{L} = 7.5\,\mathrm{k\Omega}$ was chosen, which leads to pre- and post-gains of approx. 1/4 and 4, respectively.

The proposed structure is shown in Fig. 9, in this case employing four folders. In addition to the folding stages, a saturator is placed at the output to model the behaviour of an output buffer stage. Table 1 summarizes the component and constant values for the proposed structure. Figure 8(a) shows the result of processing a unity gain sinusoidal input with this structure for $G = 10$ and zero DC offset. Figure 8(b) illustrates the outcome of processing the same waveform for $G = 10$ and a DC offset of 5V.

A real-time demo of the proposed topology implemented using Max/MSP and Gen is available at http://research.spa.aalto.fi/publications/papers/smc17-wavefolder.

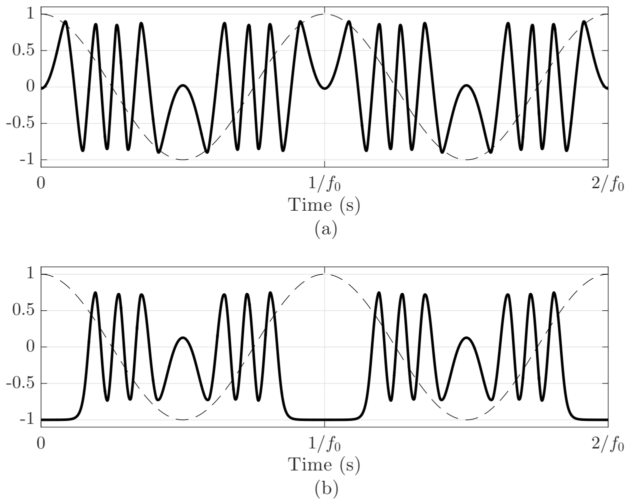


Figure 8. Waveform for a unit gain sine signal processed using the proposed cascaded structure with (a) $G = 10$ and (b) $G = 10$ plus a 5-V DC offset.

Component	Value	Constant	Value
R	15 kΩ	V_T	26 mV
R_L	7.5 kΩ	I_S	10^{-17} A

Table 1. Summary of component and constant values for the proposed cascaded model. Parameter $F_\mathrm{s} = 88.2\,\mathrm{kHz}$.

7. CONCLUSION

In this work we explored the behavior of the Lockhart wavefolder circuit, a West Coast-style nonlinear waveshaper. A VA model of the circuit was then derived using the Lambert-W function. Results obtained were validated against SPICE simulations of the original circuit. To tackle the aliasing caused by the nonlinear nature of wavefolding, the proposed model was extended to incorporate the first-order antiderivative antialiasing method. When combined with oversampling by factor 2, the antialiased wavefolder model is free from perceivable aliasing while still being suitable for real-time implementation.

Furthermore, a proposed synthesis topology consisting of four cascaded wavefolding stages was presented. The recommended structure demonstrates the capabilities of the derived circuit model in a synthesis environment. However, the proposed topology is not unique, as it can be modified according to the needs of the particular application. For instance, the number of stages can be modified. Similarly, the value of the internal load resistance can be increased for added brightness. This effectively showcases the flexibility of VA models.

Future work on the topic of wavefolding will focus on modeling the original Buchla timbre circuit. This kind of work can then extend to the study of other West Coast circuits and synthesis techniques.

Acknowledgments

The authors would like to thank Prof. Vesa Välimäki for his assistance generating the ANMR curves and for his helpful

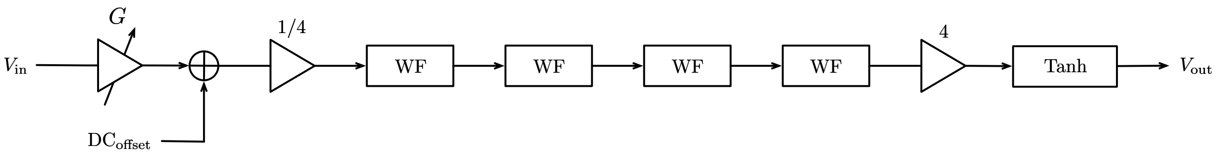


Figure 9. Block diagram for the proposed cascaded wavefolder structure. Input gain and DC offset are used to control the timbre of the output signal.

comments during the final preparation of this work. F. Esqueda is supported by the Aalto ELEC Doctoral School.

8. REFERENCES

[1] R. A. Moog, "A voltage-controlled low-pass high-pass filter for audio signal processing," in *Proc. 7th Conv. Audio Eng. Soc.*, New York, USA, Oct. 1965.

[2] Buchla Electronic Musical Instruments. The history of Buchla. Accessed March 3, 2017. [Online]. Available: https://buchla.com/history/

[3] J. Parker and S. D'Angelo, "A digital model of the Buchla lowpass-gate," in *Proc. Int. Conf. Digital Audio Effects (DAFx-13)*, Maynooth, Ireland, Sept. 2013.

[4] K. Stone. Simple wave folder for music synthesizers. Accessed March 3, 2017. [Online]. Available: http://www.cgs.synth.net/modules/cgs52_folder.html

[5] D. Rossum, "Making digital filters sound "analog"," in *Proc. Intl. Computer Music Conf. (ICMC 1992)*, San Jose, CA, USA, Oct. 1992, pp. 30–34.

[6] V. Välimäki, S. Bilbao, J. O. Smith, J. S. Abel, J. Pakarinen, and D. Berners, "Introduction to the special issue on virtual analog audio effects and musical instruments," *IEEE Trans. Audio, Speech and Lang. Process.*, vol. 18, no. 4, pp. 713–714, May 2010.

[7] T. Stilson and J. O. Smith, "Analyzing the Moog VCF with considerations for digital implementation," in *Proc. Int. Computer Music Conf.*, Hong Kong, Aug. 1996.

[8] A. Huovilainen, "Non-linear digital implementation of the Moog ladder filter," in *Proc. Int. Conf. Digital Audio Effects (DAFx-04)*, Naples, Italy, Oct. 2004, pp. 61–164.

[9] T. Hélie, "On the use of Volterra series for real-time simulations of weakly nonlinear analog audio devices: application to the Moog ladder filter," in *Proc. Int. Conf. Digital Audio Effects (DAFx-06)*, Montreal, Canada, Sept. 2006, pp. 7–12.

[10] F. Fontana and M. Civolani, "Modeling of the EMS VCS3 voltage-controlled filter as a nonlinear filter network," *IEEE Trans. Audio, Speech, Language Process.*, vol. 18, no. 4, pp. 760–772, Apr. 2010.

[11] S. D'Angelo and V. Välimäki, "Generalized Moog ladder filter: Part II – explicit nonlinear model through a novel delay-free loop implementation method," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 22, no. 12, pp. 1873–1883, Dec. 2014.

[12] A. Huovilainen, "Enhanced digital models for analog modulation effects," in *Proc. Int. Conf. Digital Audio Effects (DAFx-05)*, Madrid, Spain, Sept. 2005, pp. 155–160.

[13] D. T. Yeh, J. Abel, and J. O. S. III, "Simulation of the diode limiter in guitar distortion circuits by numerical solution of ordinariy differential equations," in *Proc. Int. Conf. Digital Audio Effects (DAFx-07)*, Bordeaux, France, Sept. 2007, pp. 197–204.

[14] J. Pakarinen and D. T. Yeh, "A review of digital techniques for modeling vacuum-tube guitar amplifiers," *Computer Music J.*, vol. 33, no. 2, pp. 85–100, 2009.

[15] J. Parker, "A simple digital model of the diode-based ring modulator," in *Proc. Int. Conf. Digital Audio Effects (DAFx-11)*, Paris, France, Sept. 2011, pp. 163–166.

[16] M. Holters and U. Zölzer, "Physical modelling of a wah-wah effect pedal as a case study for application of the nodal dk method to circuits with variable parts," in *Proc. Int. Conf. Digital Audio Effects (DAFx-11)*, Paris, France, Sept. 2011, pp. 31–35.

[17] R. C. D. de Paiva, J. Pakarinen, V. Välimäki, and M. Tikander, "Real-time audio transformer emulation for virtual tube amplifiers," *EURASIP J. Adv. Signal Process*, vol. 2011, Feb. 2011.

[18] D. Yeh, "Automated physical modeling of nonlinear audio circuits for real-time audio effectspart II: BJT and vacuum tube examples," *IEEE Trans. Audio, Speech, Language Process.*, vol. 20, no. 4, pp. 1207–126, May 2012.

[19] F. Eichas, M. Fink, M. Holters, and U. Zölzer, "Physical modeling of the MXR Phase 90 guitar effect pedal," in *Proc. Int. Conf. Digital Audio Effects (DAFx-14)*, Erlangen, Germany, Sept. 2014, pp. 153–158.

[20] C. Roads, "A tutorial on non-linear distortion or waveshaping synthesis," *Computer Music J.*, vol. 3, no. 2, pp. 29–34, 1979.

[21] M. L. Brun, "Digital waveshaping synthesis," *J. Audio Eng. Soc.*, vol. 27, no. 4, pp. 250–266, 1979.

[22] J. Kleimola, "Audio synthesis by bitwise logical modulation," in *Proc. 11th Int. Conf. Digital Audio Effects (DAFx-08)*, Espoo, Finland, Sept. 2008, pp. 60–70.

[23] J. Kleimola, V. Lazzarini, J. Timoney, and V. Välimäki, "Vector phaseshaping synthesis," in *Proc. 14th Int. Conf. Digital Audio Effects (DAFx-11)*, Paris, France, Sept. 2011, pp. 233–240.

[24] T. Stilson and J. Smith, "Alias-free digital synthesis of classic analog waveforms," in *Proc. Int. Computer Music Conf.*, Hong Kong, Aug. 1996, pp. 332–335.

[25] E. Brandt, "Hard sync without aliasing," in *Proc. Int. Computer Music Conf.*, Havana, Cuba, Sept. 2001, pp. 365–368.

[26] V. Välimäki, "Discrete-time synthesis of the sawtooth waveform with reduced aliasing," *IEEE Signal Process. Lett.*, vol. 12, no. 3, pp. 214–217, Mar. 2005.

[27] V. Välimäki, J. Nam, J. O. Smith, and J. S. Abel, "Alias-suppressed oscillators based on differentiated polynomial waveforms," *IEEE Trans. Audio Speech Lang. Process.*, vol. 18, no. 4, pp. 786–798, May 2010.

[28] V. Välimäki, J. Pekonen, and J. Nam, "Perceptually informed synthesis of bandlimited classical waveforms using integrated polynomial interpolation," *J. Acoust. Soc. Am.*, vol. 131, no. 1, pp. 974–986, Jan. 2012.

[29] E. Brandt, "Higher-order integrated wavetable synthesis," in *Proc. Int. Conf. Digital Audio Effects (DAFx-12)*, York, UK, Sept. 2012, pp. 245–252.

[30] J. Kleimola and V. Välimäki, "Reducing aliasing from synthetic audio signals using polynomial transition regions," *IEEE Signal Process. Lett.*, vol. 19, no. 2, pp. 67–70, Feb. 2012.

[31] F. Esqueda, S. Bilbao, and V. Välimäki, "Aliasing reduction in clipped signals," *IEEE Trans. Signal Process.*, vol. 60, no. 20, pp. 5255–5267, Oct. 2016.

[32] F. Esqueda, V. Välimäki, and S. Bilbao, "Rounding corners with BLAMP," in *Proc. Int. Conf. Digital Audio Effects (DAFx-16)*, Brno, Czech Republic, Sept. 2016, pp. 121–128.

[33] J. Parker, V. Zavalishin, and E. Le Bivic, "Reducing the aliasing of nonlinear waveshaping using continuous-time convolution," in *Proc. Int. Conf. Digital Audio Effects (DAFx-16)*, Brno, Czech Republic, Sept. 2016, pp. 137–144.

[34] S. Bilbao, F. Esqueda, J. Parker, and V. Välimäki, "Antiderivative antialiasing for memoryless nonlinearities," *IEEE Signal Process. Lett.*, to be published.

[35] R. Lockhart Jr., "Non-selective frequency tripler uses transistor saturation characteristics," *Electronic Design*, no. 17, Aug. 1973.

[36] K. J. Werner, V. Nangia, A. Bernardini, J. O. Smith III, and A. Sarti, "An improved and generalized diode clipper model for wave digital filters," in *Proc. 139th Conv. Audio Eng. Soc.*, New York, USA, Oct.–Nov. 2015.

[37] A. Bernardini, K. J. Werner, A. Sarti, and J. O. S. III, "Modeling nonlinear wave digital elements using the Lambert function," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 63, no. 8, pp. 1231–1242, Aug 2016.

[38] R. C. D. de Paiva, S. D'Angelo, J. Pakarinen, and V. Välimäki, "Emulation of operational amplifiers and diodes in audio distortion circuits," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 59, no. 10, pp. 688–692, Oct. 2012.

[39] D. Veberič. Having fun with Lambert W(x) function. Accessed March 3, 2017. [Online]. Available: https://arxiv.org/pdf/1003.1628.pdf

[40] C. Moler. The Lambert W function. Accessed May 19, 2017. [Online]. Available: http://blogs.mathworks.com/cleve/2013/09/02/the-lambert-w-function/

[41] H.-M. Lehtonen, J. Pekonen, and V. Välimäki, "Audibility of aliasing distortion in sawtooth signals and its implications for oscillator algorithm design," *J. Acoust. Soc. Am.*, vol. 132, no. 4, pp. 2721–2733, Oct. 2012.

Musical Approaches for Working with Time-Delayed Feedback Networks

Daniel Bisig
Institute for Computer Music
and Sound Technology
Zurich University of the Arts
daniel.bisig@zhdk.ch

Philippe Kocher
Institute for Computer Music
and Sound Technology
Zurich University of the Arts
philippe.kocher@zhdk.ch

ABSTRACT

Highly recurrent networks that exhibit feedback and delay mechanisms offer promising applications for music composition, performance, and sound installation design. This paper provides an overview and a comparison of several pieces that have been realised by various musicians in the context of a practice-led research project.

1. INTRODUCTION

Over three years, the ICST Institute for Computer Music and Sound Technology has run a research project entitled *FAUN – Feedback Audio Networks* that explored the musical potential of time-delay and feedback mechanisms within densely interconnected recurrent networks [1]. Such networks are interesting for several reasons: The changing activity of the network can be rendered audible through direct audification which abolishes the need for a potentially arbitrary sonification mapping. The manipulation of delay times provides a formalism of potential use for both sound synthesis and algorithmic composition. The networks lend themselves for generative approaches that explore the sonic potential of self-organised processes.

The *FAUN* project follows a practice-led approach to study how time-delayed feedback networks can be adopted for musical creation. This paper considers how musicians appropriate and integrate the conceptual, functional and aesthetic principles of these networks into their own creative practice. Accordingly, the paper does not provide a systematic analysis of particular network algorithms but rather highlights and discusses the diversity of individual approaches that were chosen by the musicians.

2. BACKGROUND

This section presents a brief overview of the application of feedback and delay principles in sound synthesis. It also provides information about approaches that transfer computational data and processes into experienceable representations. These approaches can inspire musicians to render sound synthesis techniques perceivable not only sonically but also through the tangible, spatial and interactive characteristics of sound installations.

2.1 Feedback and Delay in Sound Synthesis

Feedback and delay mechanisms play a prominent role for the creation and processing of digital audio signals [2]. In digital signal processing, typical applications include the design of recursive filters or the simulation of room acoustics (see e.g. [3]) In sound synthesis, several physical modelling techniques, such as digital waveguide synthesis, employ feedback and delay to simulate the propagation of acoustic waves through physical media (see e.g. [4]).

Approaches that employ feedback and delay mechanisms within highly recurrent networks are much less common. Such networks can give rise to interesting phenomena of self-organisation. The term *generative audio system* is used to designate generative approaches in computer music that do not operate on symbolic data but rather create the sonic output through direct audification [5].

There exist a few examples that employ neural networks for sound synthesis [6, 7]. Approaches that are not related to neural networks are equally relevant. The computer music environment *resNET* represents an early example [8]. It permits the realisation of networks for sound synthesis that consist of interconnected exciter and resonator units. More recently, a sound synthesis system was realised based on iterative maps whose variables are coupled via a network [9]. Recent research has been conducted on feedback networks consisting of time-varying allpass filters [5].

2.2 Exposure of Generative Systems

Due to their complex dynamics, time-delayed feedback networks can be employed as generative systems that exhibit autonomous and self-organised behaviours. This renders these networks attractive within the context of generative art [10, 11]. An ongoing debate in generative art refers to the challenge of devising an artwork in such a way that the specific characteristics of the underlying algorithms manifest themselves as principal aspects of the work. By directly exposing the algorithms in the perceivable characteristics of an artwork, the audience becomes engaged not only on an aesthetic level but can also gain through a process of experiential reverse-engineering an intellectual appreciation of the work [12]. The principle of rendering generative processes directly perceivable can serve as a compositional strategy in computer music [13, 14]. In this context, the method of mapping abstract algorithms into perceptible outcomes is important but also controversial [15]. By reducing the disjunction between algorithms and the sonic material they organise, the correspondence between formal and aesthetic principles can be made more compelling. In the most extreme case, there exists a full match between formal and perceptual properties of a generative system. Such a situation has been described as natural mapping [11] or ontological alignment [15].

3. WORKS

Several musicians have been invited to realise a work. Apart from the requirement to employ time-delay and feedback mechanisms as main means for sound creation, the musicians were free in their choice of musical ideas and technical implementations. The works include compositions, improvised performances, and sound installations. All works have been shown to the public during a small festival which was organised in October 2016 at the Zurich University of the Arts. An online video trailer provides a brief impression of the festival activities [1] For each work, a representative video excerpt is available online: Roj [2] , Twin Primes [3] , Errant it is [4] , Circular [5] , Thread [6] , Stripes [7] , Sheet Music [8] , Studie 24.2 [9] , Watchers [10] .

To facilitate the comparison between the various works, we standardise the description and depiction of the formal aspects. For each work, the algorithms are described by highlighting the characteristics of the network nodes, connections and topologies. Figures showing the internal characteristics of nodes and connections depict a single node as circle and a single outgoing connection as outlined rectangle. The depicted number of incoming and outgoing signal connections is not representative of the actual network topology. Fig. 1 shows a set of graphical symbols that represent common processing elements within a node or connection. Figures showing the network topology depict network nodes as circles and unidirectional network connections as black arrows. Arrows pointing on both sides are a graphical simplification of two unidirectional connections. Physical audio lines connecting the networks' output or input to loudspeakers and microphones, respectively, are depicted as bold black lines without arrowheads.

3.1 Concert Performances

3.1.1 Roj

The piece *Roj* by Bojan Milosevic employs network-based algorithms for creating a body of sonic material that was later manually arranged and performed as tape music. A core musical interest of the musician concerns the acoustic effects that result from continuously changing delay times and network topologies. For this purpose, the musician developed a network implementation that allows to dynamically change the connectivity among network nodes and the mapping of control values on delay times.

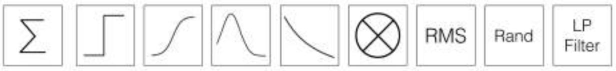


Figure 1. Graphical symbols used for the schematic representation of network nodes and connections. From left to right: summation, threshold function, sigmoid transfer function, gaussian transfer function, negative exponential function, gain function, root mean square (RMS), random number generator, low pass filter.

The node and connection characteristics are depicted in Fig. 2. Each node sums the incoming audio signals and passes the result through a gain function before sending it out to other nodes. This gain function forms part of an automated signal amplitude control mechanism. Sine oscillators or noise units can be added whose output is mapped on the delay time.

The left side of Fig. 3 shows a schematic representation of a network. While this network represents an arbitrary example, it nevertheless highlights several key elements that were used for the creation of musical material. The network possess recurrent connections that connect different nodes with each other and single nodes back on themselves. Also, the network incorporates several number generators for automating changes in delay times. The acoustic output passes through a mixing desk which permits to distribute the output spontaneously during a performance.

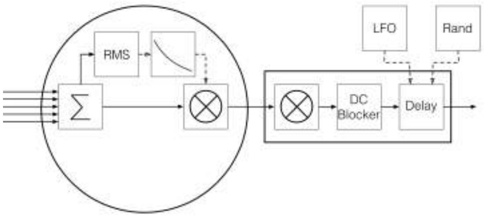


Figure 2. Node and connection characteristics for the piece *Roj*.

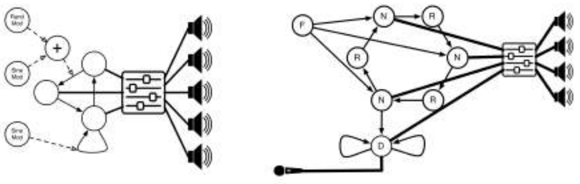


Figure 3. Network topologies for the pieces *Roj* (left) and *Errant it is* (right) . The labels stand for: F = Fourses unit, N = Izhikevich neuron, R = Karplus-Strong resonator, D = feedback and delay unit.

[1] https://vimeo.com/217825545
[2] https://vimeo.com/217974487/5fc155cb32
[3] https://vimeo.com/217974854/0ec449eac6
[4] https://vimeo.com/217974122/4ac141f4c3
[5] https://vimeo.com/217973648/30df6779f8
[6] https://vimeo.com/217851901/f8a43e9e1b
[7] https://vimeo.com/121604263
[8] https://vimeo.com/121603682
[9] https://vimeo.com/217859390/a32a315736
[10] https://vimeo.com/217893895/ad977f1a11

3.1.2 Twin Primes

The piece *Twin Primes* by Philippe Kocher combines the audio output of time-delayed feedback networks with instrumental sounds from two percussion sets. The piece establishes a relationship between musicians and network by using the acoustic output of the musicians as input material for the network. The gradual change of this relationship forms part of the musical development of the piece. Initially, the network briefly processes the percussive sounds before falling silent again, later on, the network assumes a more autonomous role and produces sustained sounds.

The node and connection characteristics are schematically depicted in Fig. 4. The node contains the same auto gain mechanism that was used in the piece *Roj*. The scaled input signal is passed through a gating element whose opening and closing is controlled by the amplitude of the microphone input: The gate opens if the RMS of the microphone signal lies above a threshold. The gate closes either after a predefined duration or when the microphone input exceeds the threshold a second time. The opening and closing of the gate is not instantaneous and its speed constitutes an additional musical control parameter. Whenever the gate closes, it triggers a randomisation of the delay values of the node's outgoing connections.

The piece uses multiple network topologies which are used either in parallel or sequentially throughout the performance (see Fig. 5). These networks receive audio input from contact microphones which are taped to various instruments of the percussion set. The acoustic output of the network nodes is routed to five speakers on stage according to a fixed matrix. The switching of networks, the control of network behaviours and parameters, and the routing of the microphone output are all predetermined as part of the compositional structure of the piece.

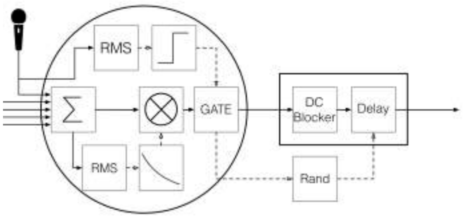


Figure 4. Node and connection characteristics for the piece *Twin Primes*.

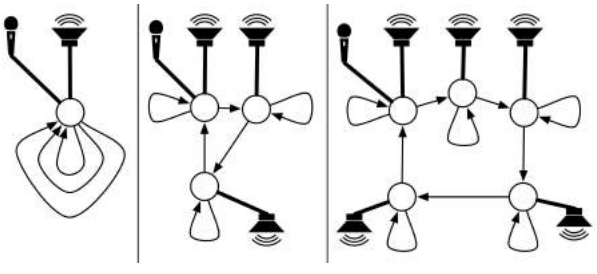


Figure 5. Example network topologies and audio routings for the piece *Twin Primes*.

3.1.3 Errant it is

The piece *Errant it is* by Volker Böhm combines a computational model of spiking neurons with several audio signal processing elements. This approach integrates the hardly predictable dynamics of a biologically inspired model network with more familiar and easily controllable elements from digital signal processing. The resulting heterogenous network serves as an improvisation tools for live performance. This network integrates the following elements: nodes that implement the *Izhikevich* spiking neuron model [16], *Karplus-Strong* string resonators, a time-delay and feedback network, and a so-called *Fourses* unit. This unit creates audio signals by confining multiple sawtooth waves within each others' amplitude envelope. [11] During the performance, various network parameters are modified in realtime using a Midi controller. Initially, the musical output is created solely from the acoustic output of the model neurons. Later on, the output of the *Fourses* unit is used to create rhythmic patterns. Towards the end, the musical output is generated by the feedback and delay network which receives its audio input from a microphone.

A graph representation of the differential equation set that describes the behaviour of the *Izhikevich* neurons is shown in Fig. 6. The characteristics of the nodes and connections that were used in the feedback and delay network is shown in Fig. 7. The nodes pass the summed input signal into an output connection. The connection contains a DC blocker, a pitch shifter, an equaliser, and a delay unit. The RMS of the node's audio signal can be used to automatically control the delay time. The topology of the heterogenous network that was used during the performance is shown on the right side of Fig. 3. The network is divided into two subnetworks. One subnetwork consist of three *Izhikevich* model neurons and three *Karplus Strong* string resonators which are connected to each other in a circular arrangement. The other subnetwork consists of a single node that possesses two recurrent delay connections to itself. A single *Fourses* unit is connected to all *Izhikevich* model neurons and serves as an external signal generator. The acoustic output of the *Izhikevich* model neurons and the time-delay feedback network node are mixed and then routed to four loudspeakers.

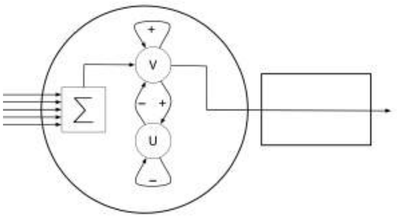


Figure 6. Node and connection characteristics of *Izhikevich* neuron models for the piece *Errant it is*.

3.1.4 Circular

The piece *Circular* is different from the other pieces in that it employs a physical feedback mechanism. The main

[11] http://vboehm.net/2014/12/fourses/ (accessed 2017-02-26)

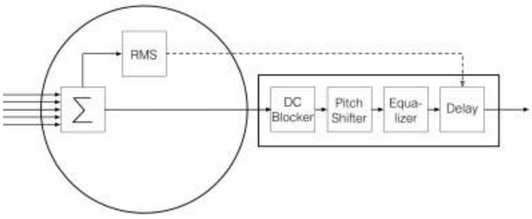


Figure 7. Node and connection characteristics of a feedback and delay network for the Piece *Errant it is*.

role of the performer is to control the feedback effects through the manipulation of a hand drum. This drum interferes with the acoustic transmission between a loudspeaker and a microphone and serves as acoustic filter and trigger. Fig. 8 shows a diagram of the stage setup on the left side and a photograph of the performance on the right side. The audio signal that is picked up by the microphone is routed into a simple signal processing patch. This patch passes the incoming audio signal through a multi band compressor and a gain unit, the latter of which can be controlled with a Midi foot pedal.

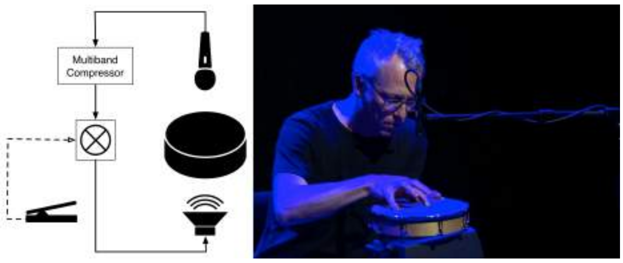


Figure 8. Diagram of the stage setup for the piece *Circular* (left) and a photograph of the performance (right). From left to right, the graphical elements represents a foot pedal and a hand drum.

3.1.5 *Thread*

The piece *Thread* employs four steel wires that serve as unconventional loudspeakers. These wires are 8 meters long and attached to two hand-rails of a car ramp. The output of multiple different feedback and delay networks which are interactively controlled by the performer is routed through transducers onto the wires. The realisation of the piece is motivated by the following musical interests: The combination of the network-based sonic material with the highly idiosyncratic acoustic properties of steel wires. A musical improvisation based on the real-time exploration of different acoustic phenomena within subregions of a large network. The performance alternates between musically minimalistic phases during which the performer hardly interferes with the network's behaviour and more musically diverse passages during which the network is subject to more frequent manual control.

The characteristics of the network nodes and connections is depicted in Fig. 9. Each node passes the sum of the input signals through a gating element before exiting through outgoing connections. This gating mechanism operates in a similar manner as has been described for the piece *Twin Primes*.

Fig. 10 shows a diagram of two example network topologies that were used during different stages of the performance. In these two networks, the node in the top left corner of each network deserves special mentioning. This node functions as compositional element in that its activity and the time delays of its outgoing connections are sequentially modified according to a predefined list of values.

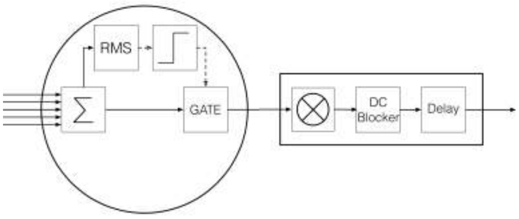


Figure 9. Node and connection characteristics for the piece *Thread*.

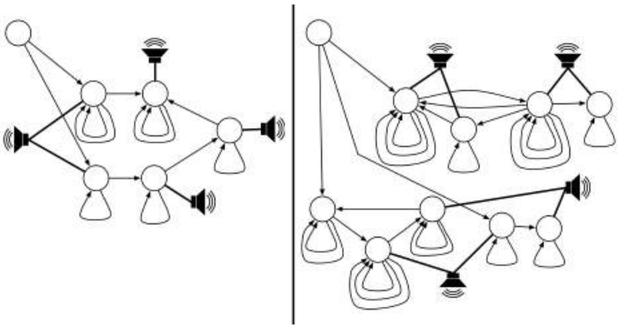


Figure 10. Example network topologies and audio routings for the piece *Thread*.

3.2 Installations

Several musicians have realised sound installations. In these installations, the acoustic output and certain aspects of the physical and/or spatial setup are set into correspondence to properties of the network algorithms. The installations presented in this section were realised by members of the ICST.

3.2.1 *Stripes*

The sonic output of the installation *Stripes* is generated by a non-standard form of waveguide synthesis that consists of a feedback system with four delay lines. Unlike regular waveguide synthesis, the delay times vary over time and they do so independently from one another. This approach permits the creation of constantly changing timbres and rhythmic gestures. The installation routes the output of each network connection to a corresponding loudspeaker. As a result, signal propagation through network connections becomes audible as a spatial propagation of sonic changes across the loudspeaker arrangement.

The characteristics of the network nodes and connections is depicted in Fig. 11. Each node employs the same auto gain mechanism that was used in the pieces *Roj* and *Twin*

Primes. The output connection contains a delay unit whose time is constantly changed by linearly interpolating between random values. The network topology is shown on the left side of Fig. 12. The network consists of a single node and four recurrent connections that connect the node with itself. The installation setup consists of four piezoelectric speaker films, each of them 1.8 metres long, that hang from the ceiling (see right side of Fig. 12).

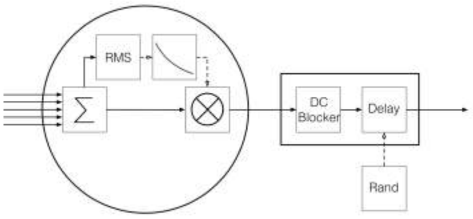


Figure 11. Node and connection characteristics for the installation *Stripes*.

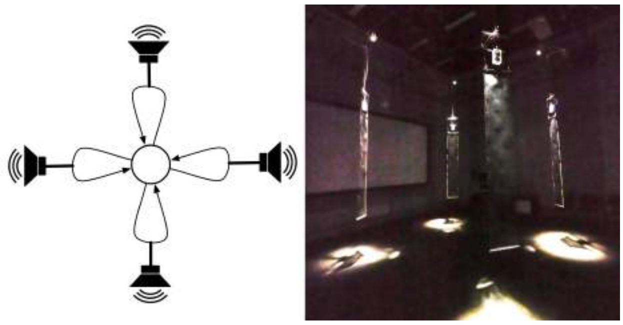


Figure 12. Network topology and audio routing for the installation *Stripes* (left) and a photograph of four piezoelectric loudspeakers (right).

3.2.2 *Sheet Music*

The installation *Sheet Music* employs a network algorithm that resembles the functioning of a switch board. Whenever a node's signal amplitude exceeds a threshold, the node's output is connected to another node. In combination with a one-to-one correspondence between node activity and acoustic output via a corresponding loudspeaker, the switching behaviour gives rise to pointillistic sonic patterns that propagate through space. In addition, the installation establishes a relationship between delay times and spatial distances among loudspeakers. The delay time between nodes is proportional to the spatial distance between the corresponding loudspeakers. This allows to exploit the spatial constraints of an exhibition situation in order to create a unique and site specific network configuration.

The characteristics of the network nodes and connections is depicted in Fig. 13. Each node passes the summed input signals through a limiter and a routing mechanism that switches in a round robin manner to its next setting whenever the node's signal amplitude exceeds a threshold. The network topology is shown on the left side of Fig. 14.

The installation setup consists of eight piezoelectric speaker films, each of them an A4 sheet in size, sitting on top of a note stand (see right side of Fig. 14). Once the installation is set up, the distances among the loudspeakers are manually measured and stored in a configuration file to be read upon initialisation of the network. A more thorough description of the installation is available in [17].

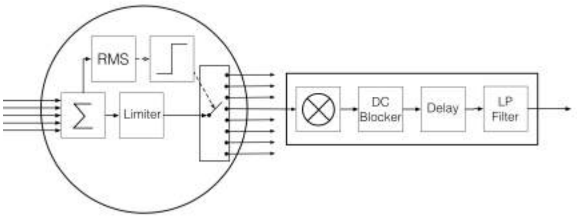


Figure 13. Node and connection characteristics for the installation *Sheet Music*.

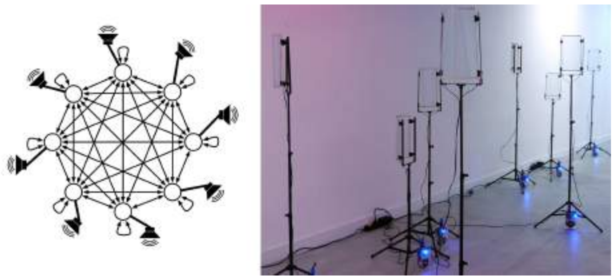


Figure 14. Network topology and audio routing for the installation *Sheet Music* (left) and a photograph of the piezoelectric loudspeakers (right).

3.2.3 *Studie 24.2*

For the realisation of the piece *Studie 24.2* the existing setup of the installation *Dodecahedron* was used. This setup possesses the shape of a platonic body and places twenty loudspeakers at equal distances from each other on the surface of a sphere. The piece appropriates the given geometric setup as compositional constraint in that it relates the physical loudspeakers and the metallic struts to nodes and connections in a feedback and delay-network. By modifying the probabilities of signal propagation across different network connections, temporary subnetworks are formed.

The node and connection characteristics are shown in Fig. 15. Here, the RMS of the signal amplitude is passed through a nonlinear gain function that boosts middle amplitudes while eliminating low and high amplitudes. In addition, a global amplitude control mechanism uses the RMS of all network signals to adjust the output gain of network connections. The gain and delay values are continuously randomised, which serves to create gradually changing timbres and helps to avoid the appearance of resonances.

Fig. 16 shows a schematic representation of the network topology on the left side and a photograph of the *Deodecahedron* installation on the right side. According to the direct correspondence between installation scaffold and network topology, each node is connected to three neighbouring nodes. Signals propagate in a branching manner, i.e.

a signal that arrives at a particular node from one of its neighbouring nodes can only be propagated to the other two neighbouring nodes. The probability and scaling of this branching propagation are randomly changed. This propagation principle leads to the appearance of local sonic movement trajectories.

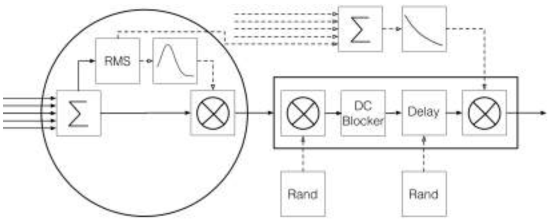


Figure 15. Node and connection characteristics for the piece *Studie 24.2*.

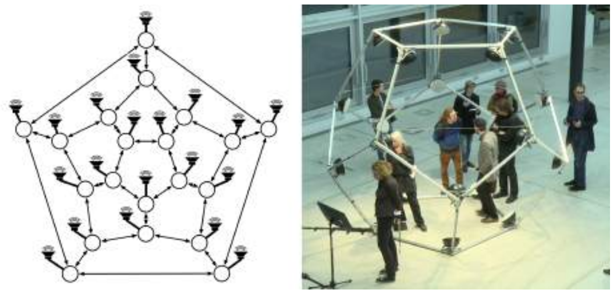


Figure 16. Network topology and audio routing for the piece *Studie 24.2* (left) and a photograph of an exhibition setup (right)

3.2.4 Watchers

The installation *Watchers* allows visitors to manually alter the horizontal orientation of loudspeakers and thereby affect the topology of the network-based synthesis system. Each loudspeaker renders the activity of a particular network node audible. Loudspeakers possess a line of sight which controls the existence of network connections. Depending on a loudspeaker's orientation, different neighbouring loudspeakers fall into its line of sight. Network connections are created between nodes that correspond to loudspeaker that can "see" each other. Whenever the establishment of a connection leads to the propagation of activity among network nodes, the sonic characteristics of the corresponding loudspeakers become gradually intermixed.

The characteristics of the network nodes and connections is depicted in Fig. 17. A node contains an internal sine oscillator whose output is combined with the sum of the incoming signals. This combined signal is passed through a sigmoid wave shaping function that serves as audio distortion and gating mechanism. The shape of the wave shaping function changes based on whether the RMS of the input signal lies within or outside a lower and upper threshold.

Fig. 18 shows a schematic representation of the network topology and audio routing. Each loudspeaker is attached to a rotational joint. Also, each loudspeaker possesses a unique identifier that is communicated to other loudspeakers vie an infrared sender mounted at the front of their casing. This signal is received by other loudspeakers via an infrared receiver that sits at the back of their casing. Each loudspeaker then transmits the list of received identifiers via Wifi to a computer which updates the topology of the sound synthesis network accordingly. A more thorough description of this installation is provided in [18].

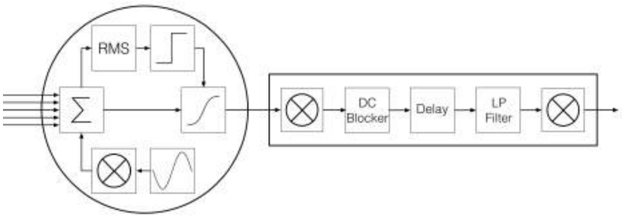


Figure 17. Node and connection characteristics for the piece *Watchers*.

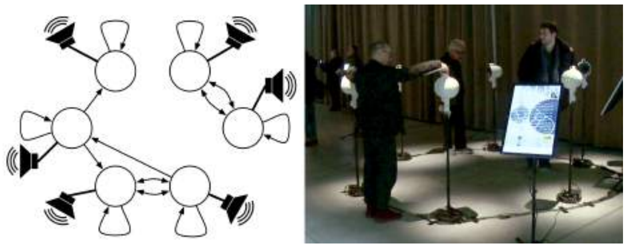


Figure 18. Network topology and audio routing for the installation *Watchers* (left) and a photograph of the exhibition situation (right)

4. DISCUSSION

This section provides a comparison and discussion of the musical approaches for the previously described pieces.

4.1 Algorithm Design

On a basic level, the design of sound synthesis network algorithms can either be informed by techniques from digital signal processing or from the appropriation of algorithms from non-musical domains. The former approach is familiar to most musicians whereas the latter results in non-standard techniques whose musical usefulness is more difficult to assess. In *FAUN*, most musicians chose a digital signal processing approach in their network implementation. Only the piece *Errant it is* implements a neural network. However, this implementation is complemented with more conventional signal processing algorithms.

4.1.1 Feedback Stabilisation

One of the challenges when working with feedback systems concerns the avoidance of instabilities caused by positive feedback. Most of the pieces employ an ad hoc approach for the automated stabilisation of feedback. The corresponding mechanisms either directly control local or global signal gain or employ a gate that closes whenever the signal exceeds a threshold. Only in two pieces for live improvisation (*Errant it is*, *Circular*), the task of feedback stabilisation is delegated to manual intervention.

Unique approaches for feedback stabilisation are employed by the following pieces: *Sheet Music* implements a connection switching mechanism that cuts of the propagation of signals that exceed an amplitude threshold, *Studie 24.2* uses a global signal attenuation mechanism and also boosts signals with average amplitude that serves to maintain a fairly constant signal amplitude throughout the piece, *Watchers* uses a wave shaping function to control signal amplitude.

4.1.2 Network Topologies

While all networks described in this paper contain recurrent connections, the characteristics and topologies of the networks are quite diverse. In all pieces with the exception of *Studie 24.2*, many or all recurrent connections connect a node back to itself. While this is obvious for networks that consist of one node only, local recurrent connections are important in larger networks as well since they allow to maintain the activity of a node for extended periods of time. In many of the networks that consist of a few nodes only (3 to 5), the recurrent connections across different nodes serve to establish circular propagation patterns. In networks containing larger number of nodes (6 to 20), circular recurrent patterns appear as well but they do not span all neurons and thereby create subnetworks. The usage of subnetworks within larger networks plays an important role in that it renders the complex feedback dynamics more manageable than in networks that are fully connected. Also, subnetworks permit to maintain concurrent sonically distinct network activities while allowing the occasional exchange of sonic material among the various subnetworks. None of the pieces makes use of a larger network in which the nodes are fully connected.

With the exception of *Errant it is*, all pieces use networks in which nodes and connections all possess the same characteristics. This is an indication that parametrical and topological variations in homogenous networks allow for the creation of sufficiently diverse sonic material so that the added complexity of heterogenous networks is not considered worthwhile by the musicians. On the other hand, many musicians employ several different network topologies, either in parallel or successively throughout the piece.

Some of the pieces use multiple networks with fixed topology (*Errant it is*, *Twin Primes*, *Thread*). By switching between these networks or by scaling the contribution of individual networks to the overall sonic result, the pieces can progress in a predictable manner through different musical sections. Other pieces use only a single network but continuously (*Watchers*, *Sheet Music*) or occasionally (*Roj*) modify its topology. This approach leads to less predictable results since the musical result depends both on the new topology and the former activity patterns that still persist after a topological change. On the other hand, the persistence of network activity helps to create a sonic continuity throughout the development of a piece.

4.2 Compositional Approaches

The use of feedback and delay principles implies that all sonic aspects of a musical piece (pitch, timbre, rhythm, etc.) become mutually interrelated. Accordingly, a musician cannot freely decide to modify each of these aspects individually. The specification of the velocity of the gating mechanism in *Twin Primes* and *Thread* results not only in spectral but also in rhythmical effects. The changing wave shapes in *Watchers* and the connection switching mechanisms of *Sheet Music* serve a similar purpose. In *Thread*, the rhythmic structure is largely predefined by the specification of delay times in connections that pass an initial activation trigger to network nodes. *Errant it is* uses an external mechanism for controlling rhythmicity: a *Fourses* ramp generator occasionally controls periodic changes in the membrane potential of the *Izhikevich* model neurons. The piece *Roj* represents a peculiar case. By manually arranging pre-recorded material, external rhythmic and formal elements are imposed on the music that are unrelated to the characteristics of a network mechanism. This approach offers the largest compositional freedom, but this comes at the cost of sacrificing some of the potential of the networks to operate as autonomous generative systems. Most musicians tried to strike a balance between compositional freedom and generative autonomy.

An interesting relationship between human performer and generative system develops in *Thread* and *Circular*. In both pieces, the musician's behaviour consists mostly of spontaneous reactions to the complex behaviour of the network. Accordingly, both the musician and the network possess agency and it is the mutual interplay between the two that shapes the development of the piece.

4.3 Experienceable Relationships

This section addresses the establishment of relationships between network algorithms and experienceable results that go beyond purely sonic considerations.

4.3.1 Spatialisation

In all pieces, the output of the network is spatialised by a multichannel speaker setup, but two different approaches are employed. Either the output of the network is routed through a mixing desk before being distributed across the loudspeaker setup, or the sonic output of each network node is directly connected to an individual loudspeaker. Only two pieces (*Roj* and *Errant it is*) follow the first approach. In a one-to-one routing of node output to loudspeaker, the spatialisation of the sonic output is predetermined to a large extend by the network's topology. Accordingly, this approach treats acoustic spatialisation as a property that is correlated with other musical aspects through the behaviour of the network. While this second approach constrains compositional freedom, it offers the opportunity to expose the network's dynamics as spatial trajectories of sonic material.

4.3.2 Physical Correspondences

The one-to-one routing of node output to loudspeaker can be understood as an ontological form of mapping between algorithmic properties and physical objects. That is, the individual nodes in a network manifest both as acoustic point sources and tangible objects. Accordingly, the speaker setup conveys information about the structure of the network that might otherwise be hidden from the audience. This mutual object correspondence is emphasised in *Stripes*, *Sheet Music*, *Studie 24.2*, and *Watchers*.

Apart from establishing a correspondence between network nodes and loudspeakers, some of the pieces also experiment with the exposure of the network's topology through physical or spatial metaphors. *Studie 24.2* uses the given physical structure of an installation as guiding principle for the network topology: each metallic strut between two loudspeakers corresponds to a connection between two network nodes. Another approach is chosen for *Sheet Music*, where the distances among loudspeaker are mapped onto delay times. This correspondence between spatial properties of physical space and temporal properties of a sound synthesis system allows the physical space to partially imprint itself on the musical characteristics.

In *Watchers*, relationships among loudspeakers are based on the principle of line of sight. Since visitors can affect the "visibility" among loudspeakers either by manually changing the loudspeakers' orientations or by blocking the infrared signals with their bodies, this principle of correspondence can be exploited as an interaction affordance.

5. CONCLUSION

This publication provides a comparison of different musical approaches for working with time-delay and feedback networks. We believe that the pieces realised in the context of this project are sufficiently diverse to inform a general evaluation of the creative potential of such networks. Nevertheless, it is clear that the chosen approaches hardly represent the full breadth of possibilities. Some of the pieces do not follow fundamentally different approaches but belong to one of several families of similar works that have emerged during the project. Furthermore, two different approaches are underrepresented in this body of works: first, network-based forms of sound synthesis that incorporate algorithms from outside the musical domain in order to experiment with non-standard forms of sound synthesis, and second, installations in which musical strategies for establishing correspondences between musical algorithms and physical properties are exploited as affordances for interactive engagement. In a future continuation of this project, we hope to be able to offer calls for work that specifically focus on these topics.

6. REFERENCES

[1] D. Bisig and P. Kocher, "Early investigations into musical applications of time-delayed recurrent networks," in *Proc. of the Generative Art Conference*, Milan, 2013.

[2] D. Sanfilippo and A. Valle, "Towards a typology of feedback systems," in *Proc. of the Int. Computer Music Conference*, 2012, pp. 30–37.

[3] D. Rocchesso and J. O. Smith, "Circulant and elliptic feedback delay networks for artificial reverberation," *IEEE Transactions on Speech and Audio Processing*, vol. 5, no. 1, pp. 51–63, 1997.

[4] S. Bilbao, *Wave and Scattering Methods for Numerical Simulation*. Chichester: John Wiley & Sons, 2004.

[5] G. Surges, T. Smyth, and M. Puckette, "Generative feedback networks using time-varying allpass filters," in *Proc. of the Int. Computer Music Conference*, 2015.

[6] K. Ohya, "Sound Variations by Recurrent Neural Network Synthesis," in *Proc. of the Int. Computer Music Conference*, Ann Arbor, 1998, pp. 280–283.

[7] A. Eldridge, "Neural Oscillator Synthesis: Generating Adaptive Signals with a Continuous-Time Neural Model," in *Proc. of the Int. Computer Music Conference*, Barcelona, 2005.

[8] M. Hamman, "Dynamically Configurable Feedback/Delay Networks: A Virtual Instrument Composition Model," in *Proc. of the Int. Computer Music Conference*, Aarhus, 1994, pp. 394–397.

[9] B. Battey, "Musical Pattern Generation with Variable-Coupled Iterated Map Networks," *Organised Sound*, vol. 9, no. 2, pp. 137–150, 2004.

[10] P. Galanter, "What is Generative Art? Complexity Theory as a Context for Art Theory," in *Proc. of the Generative Art Conference*, Milan, 2003.

[11] A. Dorin, J. McCabe, J. McCormack, G. Monro, and M. Whitelaw, "A Framework for Understanding Generative Art," *Digital Creativity*, vol. 23, pp. 239–259, 2012.

[12] M. Whitelaw, *Readme 100: Temporary Software Art Factory*. Norderstedt: Books on Demand, 2005, pp. 135–154.

[13] T. Davis, "Complexity as Process: Complexity-Inspired Approaches to Composition," *Organised Sound*, vol. 15, no. 02, pp. 137–146, 2010.

[14] S. Kim-Cohen, *In the blink of an ear: Toward a non-cochlear sonic art*. A&C Black, 2009.

[15] A. Eldridge, "The poietic power of generative sound art for an information society," *the sciences*, vol. 2, no. 7, p. 18, 2012.

[16] E. M. Izhikevich, "Simple model of spiking neurons," *IEEE Transactions on neural networks*, vol. 14, no. 6, pp. 1569–1572, 2003.

[17] P. Kocher and D. Bisig, "The sound installation 'sheet music'," in *Proc. of the Generative Art Conference, Rome*, 2013.

[18] D. Bisig, "Watchers an installative representation of a generative system," in *Proc. of the 5th Conference on Computation, Communication, Aesthetics & X, Lisboa*, 2017.

Electric grammars. Algorithmic design and construction of experimental music circuits

Simone Pappalardo
Conservatorio Statale di Musica “O. Respighi” di Latina
simpapp@yahoo.it

Andrea Valle
CIRMA-StudiUm - Università di Torino
andrea.valle@unito.it

ABSTRACT

The paper discusses a generative approach to the design of experimental electronic circuits for musical application. The model takes into account rewriting rules inspired by L-systems constrained by domain-specific features depending on electronic components, and generates families of circuits. An integrated production pipeline is introduced, that ranges from algorithmic design to simulation up to hardware printing.

1. INTRODUCTION

A generative approach has been intensely pursued in the digital domain, i.e. where a certain generative system has an established connection between software and final output. Computer music has worked extensively on algorithmic composition techniques that have been applied since its inception both to event organization and audio generation [1]. On the other side, physical computing [2] has successfully tried in recent years to fill the gap between computation and the physical world, by linking algorithmic design to a heterogeneous set of production systems, e.g. in the extensive applications of 3D printing. Concerning electronics, and in particular electronic circuit design, it can be observed that the latter is a complex task, as it involves accurate selection of components in relation to a specific desired output. Thus, algorithmic techniques are used on the design side mostly for circuit optimization. As an example, evolutionary algorithms have been used to design circuits: the rationale is that in some cases a certain performance is expected and the algorithm is iteratively run so to design a circuit to fit the targeted performance [3]. In this sense, generation is constrained by an expected result rather than focusing on the exploration of new behaviors (see also [4] for AI techniques based on an evolutionary computation for circuit design). Moreover, hardware layout for electronic components involves many critical steps, depending on both component features (packaging) and routing constraints (e.g. non overlapping of routes). Thus, a completely automated pipeline as required by a truly generative approach, linking algorithmic models to final physical output (i.e. electronic components) is a difficult task. On the other side, nowadays available technologies seem promising in favoring such an approach, as a panoply of Electronic Design Automation (EDA) software solutions are available, that can be directly linked to automatic circuit printing. In the following, we discuss a generative methodology for audio circuit design, propose an application, and describe an integrated pipeline solution from circuit modeling to PCB printing.

2. ALGORITHMIC DESIGN FOR AUDIO CIRCUIT

Generative approaches to art and creativity are typically intended as ways to yield complex structures by defining compact formal procedures. A plethora of models and techniques has been defined in the last decades, including, among the others, formal grammars, fractals, cellular automata [5]. As such approaches are formal, they have been applied/mapped to various aesthetic domains, including e.g. visual art and music composition. The rationale at their base is to shift the focus from the final work to the meta-level, that of a system capable of generating various “legal”, “well-formed” outputs. In turn, outputs may be evaluated, e.g. filtered in terms of acceptance/refusal in relation to some (un)desired features, and the evaluation may be used to tune the system by means of a feedback loop. In short, algorithmic techniques like the aforementioned ones are mostly used to explore the output space of a system. In the domain of sound production, analog audio synthesis –far from being a dead end– has survived to digital audio and it is at the moment flourishing among musicians, in particular in hybrid systems that retain analog audio generation while assigning control to digital modules. In the field of electronic circuits for music applications, modular systems that can be assembled in various ways are a standard solution since the inception of analog electronic music, and recently various efforts have been made to create miniature, highly reconfigurable modular systems (e.g. littleBits [1]). However, as the assembly is left to the user, all these systems are not suited for generative techniques. In this sense, the generative process should move to a lower hardware level. A generative approach to audio circuit design has not been investigated, as far as we know, as typical audio circuits implement well-defined schemes that are a priori constrained in relation to specific needs, e.g. noise reduction optimization or adherence to a certain already es-

[1] http://littlebits.cc

tablished model, as in the case of guitar pedal effects. On the other side, many experimental music practices focus on the manipulation of analog audio circuits with a clear exploratory aim [6]. As an extreme example, the popular practice of circuit bending [7] even proposes a paradoxical "anti-theory" as a knowledge background for a totally empirical approach to electronic hacking. Differently from the previous empirical approaches but sharing the same exploratory perspective, we propose a generative approach to audio circuit design for experimental music. The main aim of our project is to produce families of circuits, sharing some basic features, that can be printed automatically, so that each board is unique while retaining a family resemblance with the other from the same generative model.
The design methodology is a three-step process:

1. **Model retrieval**: as a first step, an already established audio circuit is taken into account, so to start with a well-know and working example;

2. **Functional decomposition**: the circuit is theoretically decomposed into functional units that can be thought as closed elements provided with inputs and outputs. These units can include various electronic component configurations. Granularity at the single component level (e.g. diode) is too basic to provide an operative base for generation, as too many constraints would have to be taken into account.

3. **Rule-based re-organisation**: functional units are considered as symbols to be arranged in strings following a specific formalization.

While conceptually separated, the three steps are tightly integrated, as there is a constant feedback loop among searching for suitable audio circuits, investigating about their fitting into a modular approach, and assessing the robustness in scaling up component assembly via generative procedure. In the following, we will discuss the previous methodology in relation to a case study.

3. MODEL RETRIEVAL AND FUNCTIONAL DECOMPOSITION IN A TEST CASE

In this section we discuss audio analog electronics, that is, in reference to the previous process, both models (1) and decomposition (2). We will discuss a specific test case to show various emerging issues, but the former is meant as an example of the previously introduced, general methodology. As already said, while it might be desirable to uncouple electronic circuit design from generative modeling, there is a feedback loop between the domain to be generatively modeled (electronic circuits) and the modeling system (see later). In general terms, our basic design choice was targeted to define an expansion procedure of a certain basic circuit. Indeed, other circuit designs may be taken into account as candidates.

Our aim has been to look for an already established audio schematic (methodology, step 1) provided with two features. First, it should be expandable by iterating a limited set of basic units, as the goal of the main electronic circuit is to link some independent object characterized by audio input and output (methodology, step 2). Second, it should provide a spectral enrichment (a music desideratum) without a final amplitude overload (an audio constraint related to generative procedure, as iteration results in stacking an undefined number of components).

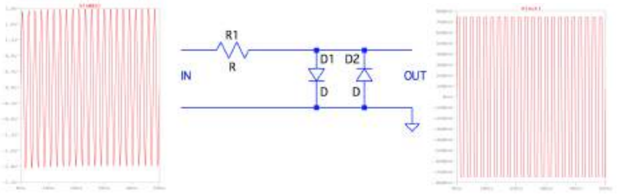


Figure 1. Diode clipping circuit.

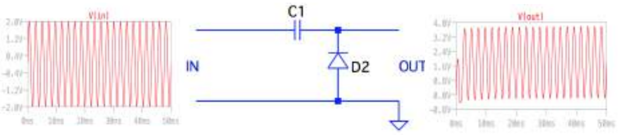


Figure 2. Diode clamping circuit.

In relation to step 1, it can be observed that diodes applications have been widely used in audio. in particular, diode clipping (Figure 1) is often used in analog and audio circuits to create distortion effects or to protect a device's input from possible current overloads: as a consequence, various schematics implementing this technique are available (e.g. [8]). Diodes used in analog effects for guitar generally clip when the input signal voltage reaches the forward threshold voltage of the component. For our circuit we decided to work with Schottky diodes so to have a very low clipping threshold (typically 0.2 V against 0.7 V of a silicon type).

In classical analog distortion guitar effects, the amplitude of the input signal is limited in either directions by a fixed amount. Indeed, in these devices, the reference threshold voltage for diodes is the circuit ground, so that the input signal clips at the small forward voltage of the forward biased diode. In practical applications, the threshold voltage of the diode, and thus therefore the signal clipping point, can be set to any desired value by increasing or decreasing the reference voltage [9]. It is therefore theoretically possible to modulate the clipping threshold of a wave by setting an audio-controlled variable threshold as the forward biasing of the diode. In our case, the threshold of the diode is controlled by a clamping circuit [10] (Figure 2). In order both to isolate the effects of two circuits (clamping and clipping) and to control impedances to implement a simulation of an audio-modulated diode with Spice3 [11], we improved the theoretical schematics in Figures 1 and 2 by introducing two operational amplifiers, to be used as voltage followers. The resulting circuit is shown schematically in Figure 3. It is still a theoretical circuit but sufficient to obtain a simulation of a "carrier" sine wave (220 Hz) dynamically clipped by another sinusoidal "modulator " (439 Hz). Figure 4 shows a new simulation of the

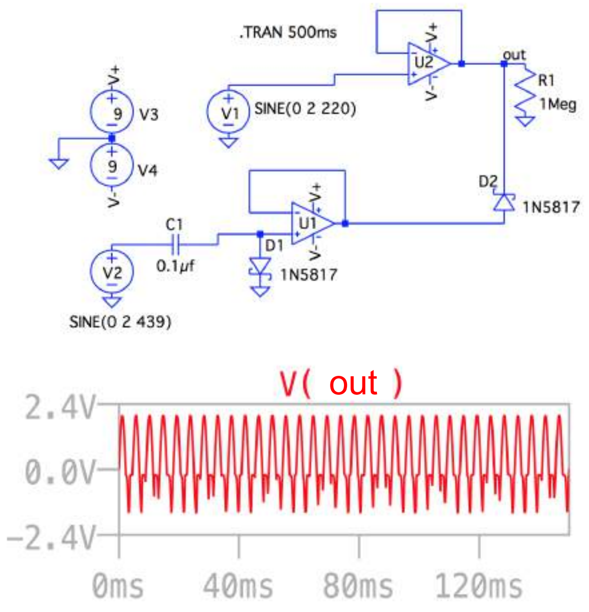


Figure 3. Half wave diode-modulated clipping (dmc)

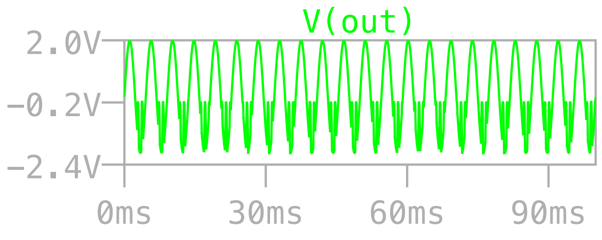


Figure 4. Half wave dmc with $C = 220$ Hz and $M = 1239$ Hz.

same schematic, this time featuring a 220 Hz carrier and 1239 Hz modulator. This schematic allows the modulation of the negative half-waves of a carrier sine by changing at audio rate the voltage reference at the anode of a diode. In order to modulate both negative and positive parts of the carrier sine wave, Figure 5 shows the implementation of a duplication/reversal of the clamping schematic. Such a clamping circuit is intended to limit a third modulating wave in the positive range. This signal is used to induce a fluctuation in the voltage at cathode of a further diode that modulates the positive part of the carrier half-wave.

Figure 5 shows a Spice3 simulation with positive modulator $=$ 1123 Hz, negative modulator $=$ 439 Hz, and carrier $=$ 220 Hz.

The outlined schematic can be expanded by a systematic development, through a nesting and multiplication of modulating objects around the axis of the carrier signal, thus fulfilling the requirement of step 2 of our methodology, that prepares rule-base organization (step 3). In order to allow the nesting of elements, we took advantage of typical features of operational amplifiers. These components (op amps, as commonly referred), thanks to their impedance features and their ease of use, allow to simply cascade several parts, adding the possibility of frequency control by

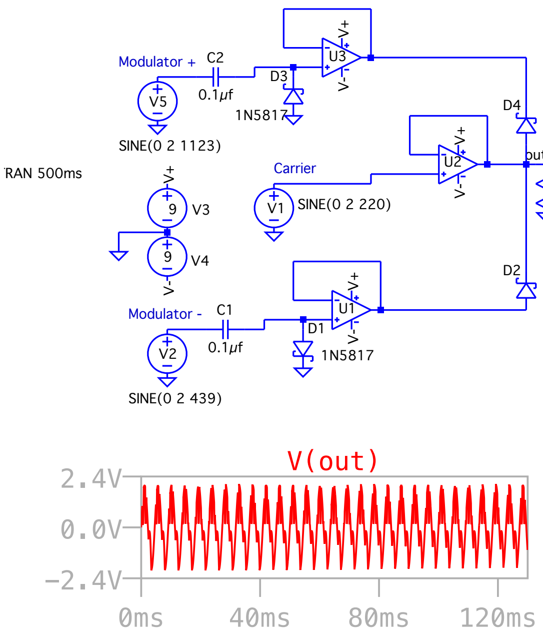


Figure 5. A modified version for full wave modulation dmc.

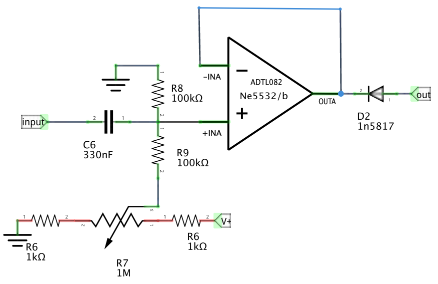


Figure 6. a) dc+.

means of inserting few capacitors. In our schematic model we add both a voltage follower object to use for coupling the various circuit each other, and an op amp input amplifier to filter, amplify and/or attenuate three input signals: carrier, negative modulator, positive modulator. The schematic model can be, then, decomposed into five units (see later): dc+, dc-, buffer, sig in, sig out.

Figures 6 to 10 show all the schematics of the final single objects. Figure 11 shows the complete root circuit ready for a Spice simulation. Colored blocks represent previously discussed components, that is, a → dc+; b → dc-; c → buffer; d →sig in; e → sig out (for other letters, see Section 4, and the companion Figure 13). Figure 12 shows a physical test implementation of the “root” circuit, a minimal assemblage of the components that acts as starting state for generation.

4. RULE-BASED RE-ORGANIZATION

In general terms, many formalisms are available as generative models to govern an arrangement of audio components. In our case, we have explored L-systems [12]. L-

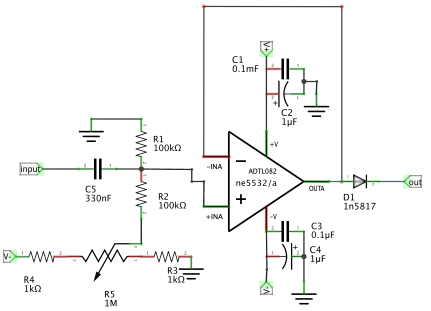


Figure 7. b) dc-.

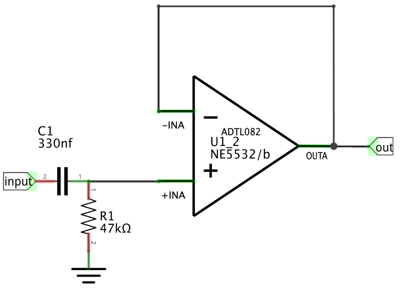


Figure 8. c) buffer.

systems have been proposed in biology as formal models to describe natural shapes that exhibit a clear generative behavior, i.e. an accumulation of features resulting from a growing pattern. They have originally proven successful in modeling plants, and then inspired algorithmic approaches to biological morphologies (shells, sponges, seaweed) [13] [14]. As formal systems, and apart from their interpretation in the biological domain, they have been widely used in generative art, including algorithmic music composition. In relation to our case, the choice of L-systems has been driven by considering an electric circuit, if not a tree, at least as a sort of vegetable rhizome. Hence the idea of recursively expanding a basic topology into a more complex network. Analogously to what happened in the model retrieval step, other formalisms may be explored.

L-systems are formal grammars in which a set of parallel rewriting rules ("production rules") is applied against an initial sequence of symbols (the "axiom"). Production rules allow to generate new sequences of symbols against which the former can be applied again. Even if the resulting morphology depends on the interpretation of symbols in a certain domain, the formal system structurally encodes features of a recursive organization. The following model for audio analog electronic circuit is inspired by L-systems. It can be defined as

$$E = (V, \omega, P)$$

where

$$V = \{a, b, c, d, e\}$$

is the set of symbols,

$$\omega = d*@1\ d*@2\ d*@3\ c1@4\ a2@4\ b3@4\ c4@5\ e5@*$$

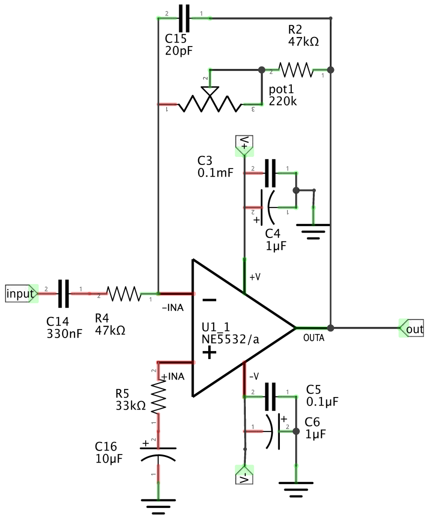


Figure 9. d) sig in.

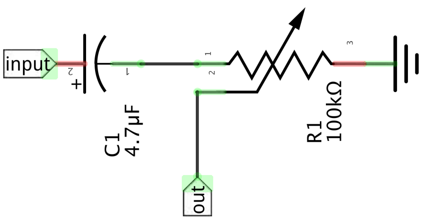


Figure 10. e) sig out.

is the axiom, and P :

$$a \to c3@(k+1)\ b2@(k+1)\ c(k+1)@(k+2)\ a(k+2)@4$$
$$b \to c2@(k+3)\ a3@(k+3)\ c(k+3)@(k+4)\ b(k+4)@4$$

is the production ruleset.

As a variation on standard L-systems, the proposed model includes integers associated to each symbol in the form $i@o$, where @ is a syntactic separator. The symbol k represents an integer associated to the complete rewriting n (i.e. application of all rules), so that $k = n \times 4$. Empty spaces and brackets have only a typographical meaning for sake of readability. In the interpretation context, only the string resulting from the last rewriting is taken into account. Each symbol from V is associated to the relative electric component (see Section 3). The symbols i and o are interpreted as labels respectively for input and output ports. Thus, a symbol like $b2@14$ indicates that the component b has an input port labeled 2 and an output port labeled 14. The symbol $*$ represents a null value: it can be seen that components d have a null input, as they are signal generators that feed the circuit, while e has a null output, being itself the output of the system. In order to define the topology of the circuit, port labels are taken into account, and input ports are connected to output ones with the same label, the signal flowing from output to input. Graphical interpretation of the strings allows to inspect visually the resulting electronic topologies. The following graphs have been obtained by automatically scripting the dot language for graph visual-

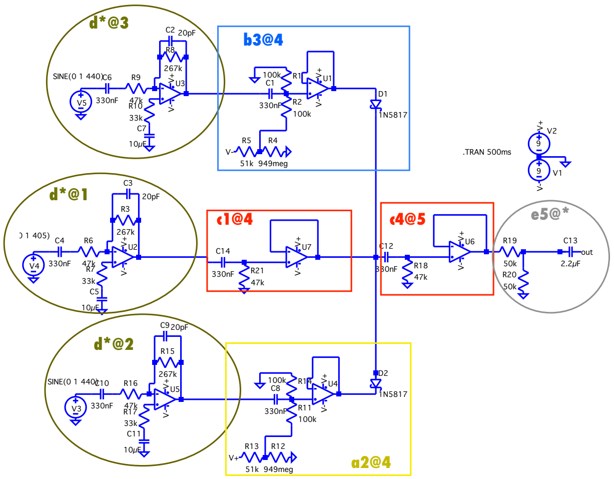


Figure 11. Root circuit (axiom).

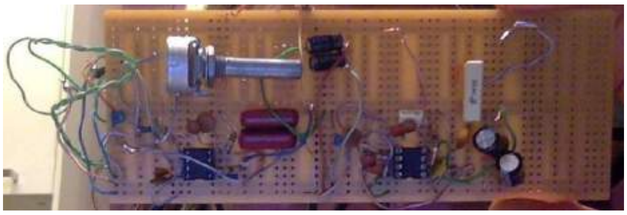


Figure 12. Root circuit, first test.

ization [15]. Figure 13 shows the topology of the circuit of Figure 11 in relation to its modeled decomposition into the axiom ω. Each component is colored in relation to its type, and labeled according to the mapped symbol, like in Figure 11. It can be seen that output ports are connected to input ports with the same identifier. Figure 14 shows the first application of the ruleset ($n = 1$).

The definition of the formal system has been driven by semantic constraints, i.e. by taking into account viable electronic connections. While increasing n, the resulting string shows some spurious artifacts, in terms of its electronic interpretation. Figure 15 shows the topological graph with $n = 2$ (labels have been replaced with the component types). It is apparent from Figure 15 that buffers at the graph' side are misplaced in the electronic interpretation, as they are isolated. To solve this issue, a recursive pruning strategy is applied to the graph, so that no isolated buffers are present in the final graph. Figure 16 shows the graph of Figure 15 after pruning.

5. AN INTEGRATED PIPELINE

While algorithmic modeling of electronic circuit topologies can have a theoretical interest, it is indeed very far from a practical application. The aim of a generative approach is indeed to obtain circuits whose complexity cannot be reached by hand design and that are (o should be) still guaranteed to work by the system definition itself. This requires an integrated automatic pipeline, ideally from formal model seamlessly to PCB printing. The implemented solution is shown in Figure 18. Grey boxes represent used softwares, while white boxes are internal blocks developed to perform specific functions. Plain text labels represent textual data. The generative model has been implemented in SuperCollider [16], an audio-targeted programming language that features high-level data collection manipulation. The SuperCollider component performs two task: generation and mapping. L-Generator is the module implementing the L-system generation: it takes an L-system specification as a textual input, and output a string (as a result of rewriting). The string is fed into the Parser that converts it into a data structure apt to be further manipulated by the Pruner module. The resulting data structure can then be mapped into other formats. The mapping task can be seen as a software gluing process. The DotGenerator creates a dot file to be subsequently rendered by the dot program as a visual file. As discussed, this is instrumental in rapidly understanding the behavior of the generation process. LTGenerator creates input files (in the “asc” ASCII format) for the LTSpice IV software [17]. LTSpice IV includes simulation of the circuit via Spice3 (including audio generation, a crucial features in our case) and schematic drawing. In order to generate asc files, component descriptions are stored in specification templates (spec file), in which opportune textual replacements for identifiers and positions are performed. Component circuits (a–e) are algorithmically placed on the board by following a simple carriage return strategy over a fixed width grid and using textual labeling of input and output as internal ref-

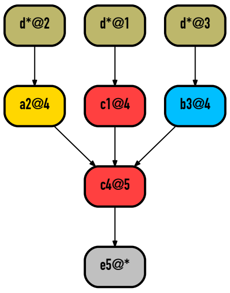


Figure 13. Axiom, with connection labels.

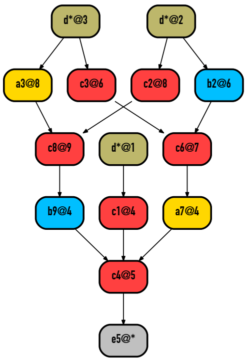


Figure 14. First rewriting, with connection labels.

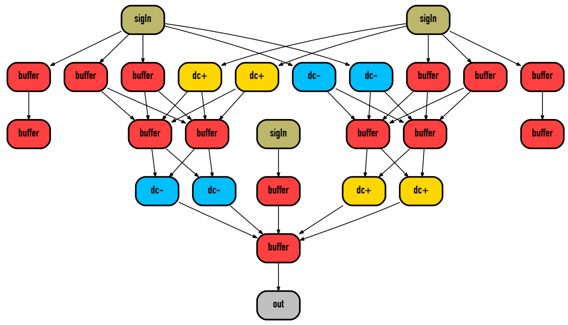


Figure 15. Second rewriting.

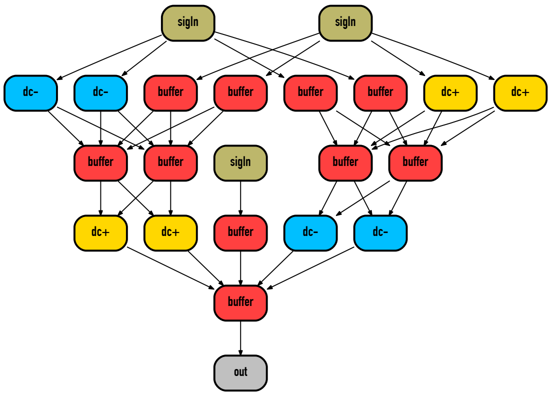


Figure 16. Second rewriting, pruned.

erences. Figure 23 shows the circuit of Figure 17, as converted by LTGenerator into asc format and drawn by LTSpice VI, with a zoomed component for sake of readability. Figures 20 and 19 show sonograms and waveforms from Spice 3 audio simulation of a very large circuit, obtained by four rewritings. It can be seen that complex spectra are easily obtained, with many partials in various harmonic relations. Spectra are sensitive to small parameter variations, thus ensuring a varied palette that can be exported as a user control (e.g. in the case of the three modulating frequencies, the latter can be associated to rotary controls on the final board). While Figure 19 illustrates a fixed spectrum, Figure 20 shows an evolving one, resulting from real-time parameter tweaking. While these results from extensive tests –matching the typical flavor of analog audio modulation but in a very rich way– are very encouraging, the main (in fact, surprising) drawback of simulation is that it has proved to be far from accurate (see later), so a real test phase has to be performed on the final hardware circuits. The asc format is also used as an interchange format to communicate with EasyEDA [2] , a web-based EDA tool suite. EasyEDA has functionalities to facilitate (even if not to fully automate) routing for printed circuit board layouts and to directly manufacture printed circuit boards. In this way, it is possible to link the formally generated specification file to PCB printing. Figure 22 shows the root circuit PCB (i.e. the axiom) obtained by running the autorouting procedure in EasyEDA.

6. OPEN ISSUES

At the moment of writing, we have tested physical circuits on breadboard, generated layout diagrams, run extensive simulations, while we have still not printed PCBs. Thus, we cannot evaluate the very final product of the pipeline. In any case, two main issues have emerged.

Simulation vs. Reality. For each object, various tests were made in order to find the best configuration in terms of matching between Spice3 simulation results and real physical behavior (tested on breadboard implementations). This is a crucial point, as –in order to simulate with a good approximation the circuit even after several expansions by the generative system– it is essential that the Spice simulation is very close to the actual behavior of the circuit. Otherwise, unpredictable, non-functional results may emerge in the printed circuit. As a matter of facts, real circuits assembled on breadboard have always shown relevant differences in their behavior if compared to Spice3 simulations. Three main factors account for these differences:

– the power supply quality which must be opportunely and recursively filtered, unlike in the simulated schematic;

– the extensive need, between input and output of several op amp, for decoupling capacitors, not necessary (at least in the same amount) in the simulated circuit;

– the placement of the components on the board, that may create ground loops, again absent from Spice.

The physical circuit is usually more sensitive to small, sometimes timy, variations in the starting conditions. Here, nonlinearity of the components seems to be more visible. For example, in our case, small variations in the signal amplitudes (modulating and/or carrier) result in large differences in the resulting output. The simulation in Spice seems to smooth out the nonlinearity of the components. In this sense, the diode clamping (circuits a-dc+ and b-dc-) is the most sensitive to small variations of the input signals. We tried physically different circuits to limit and optimize the results. in order to allow a more flexible modulation of the threshold of the diode we chose to use a circuit that allows us to select by means of a potentiometer the amount of dc-offset output. A different amounts of offset corresponds to a greater or lesser depth of action of the modulating diode.

Pipelining. Figure 21 shows an ideal, 5-step model. First, an abstract topology of the circuit, such as the ones in Figure 14-16, is generated (1). Then, the topology is mapped onto an actual electronic topology, incorporating data from real electronic components, but without any metrics referred to the final circuit layout, so that a simulation can be rendered (2). The electronic circuit should be rendered as a schematic in order to provide a standard visual feedback on the result (3). Finally, the circuit schematic should be converted into a description of the physical layout, including component packaging and routes (4), so to be delivered for PCB printing (5). Candidates for step 1 include substantially all available programming languages, better if shifted towards high-level due to data manipulation in the formal

[2] https://easyeda.com/

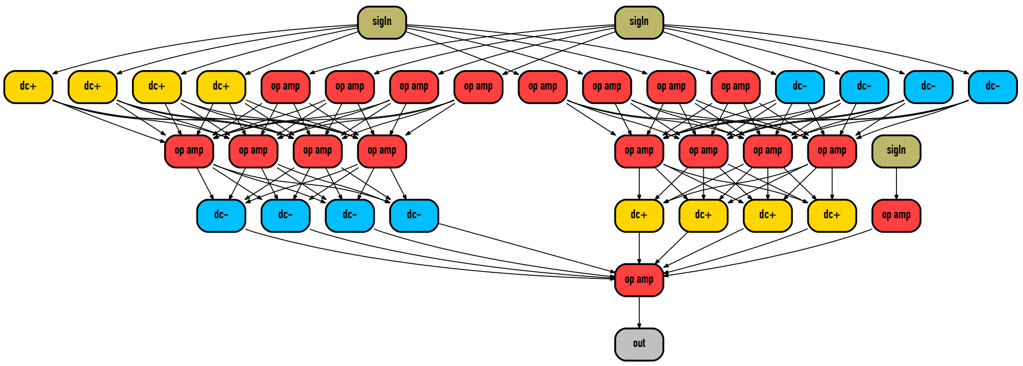


Figure 17. Third rewriting, pruned.

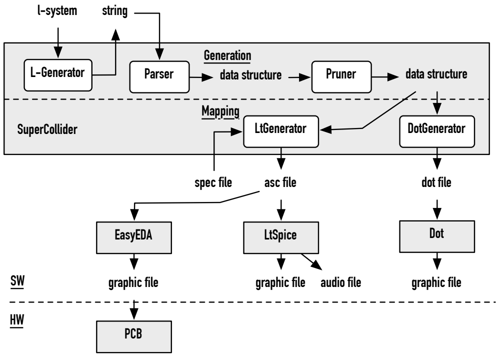


Figure 18. Implemented integrated pipeline.

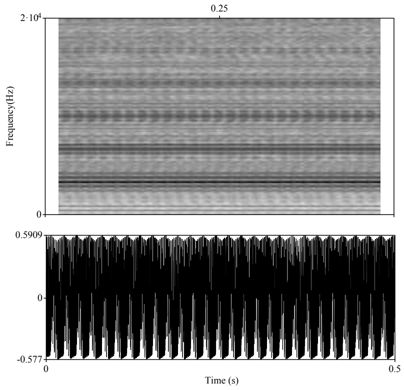


Figure 19. LTSpice audio simulation with 4 rewritings, $55, 44, 3345.7$ Hz.

model. A standard candidate for step 2 is indeed Spice3, that takes as its input a so-called netlist, i.e. a textual specification of components and connections, and is able to run simulation (including audio) and perform various analysis. Various software packages are available for steps 4 and 5, like e.g Autodesk Eagle. Actual available softwares for electronic layout are intended as computer-aided software for interactive editing toward the final layout organization, not as automated tools. A critical issue in relation to the ideal pipeline is that all the data should automatically transit from one step to the other. As an example, our choice (EasyEDA) has been determined by file format compatibility with LTSpice IV. On one side, schematic capture softwares (as required by 4) are intended for design, and includes Spice3 as a simulation engine, so that typically they are able to export netlist files but not to import them. This depends on the fact that such a passage would require automatic layout for drawing circuit schematic. In our case, we solved by defining a grid algorithm. On a different but correlated side, EDA softwares (5) aim at computer-aided layout design, but not at a fully automatic one. Autorouting facilities are available, but, as final physical layout depends on such a large number of parameters, automation is only partially possibles. Of course, very large, automatically generated circuits still require a large amount of hand-operated fine tuning of the layout before printing.

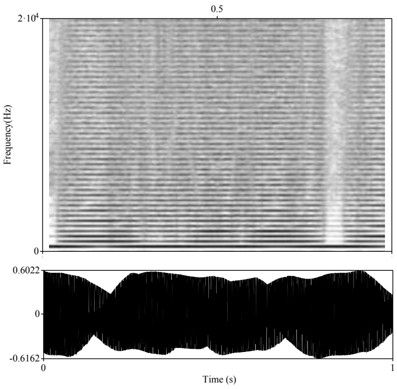


Figure 20. LTSpice audio simulation with 4 rewritings, $439, 440, 442.5$ Hz.

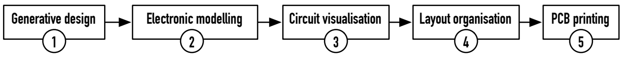


Figure 21. Ideal pipeline.

7. CONCLUSIONS

While still at a prototypical stage, we believe that our approach has shown potential interest for experimental computer-design of audio analog synthesis, as can be seen from resulting signals in Figure 20 and 19. Generative models allow to describe not circuits, but families of circuits, that share some features but may differ one from each other by details. Thus, with automatic PCB routing, it could be pos-

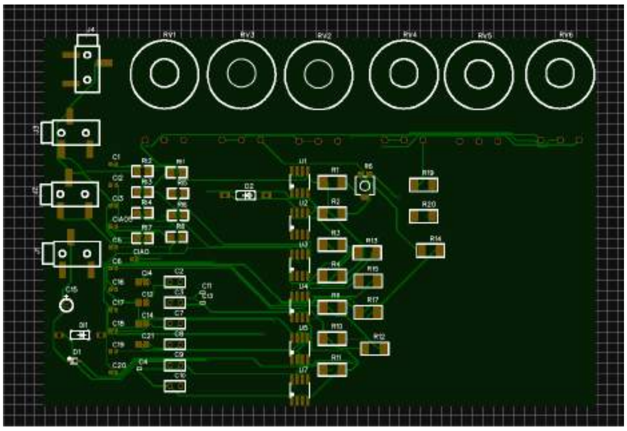


Figure 22. PCB for root circuit via autorouting.

sible to create singleton, unique synthesizers/processors. Indeed, the most delicate point is the layout for PCB printing of large generated circuits, that is still not robust from an automation perspective. But, once fine-tuned in relation to real physical behavior, circuits can be simulated, and only the final step (5) requires handcrafted operations. As a future work, we are planning to print PCB boards and test real final complex circuits, to analyze other analog circuits in order to decompose them, so that other generative models may be taken into account.

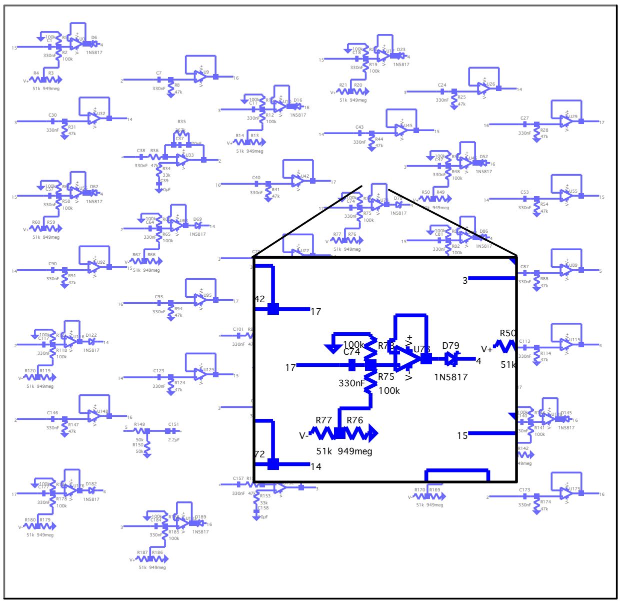


Figure 23. LTSpice drawing from automatic generated asc file, third rewriting, with a zoomed module.

8. REFERENCES

[1] C. Roads, *The Computer Music Tutorial*. Cambridge, MA, USA: MIT Press, 1996.

[2] D. O'Sullivan and T. Igoe, *Physical Computing. Sensing and Controlling the Physical World with Computers*. Boston, Mass.: Course Technology, 2004.

[3] M. M. B. Vellasco, R. S. Zebulum, and M. A. Pacheco, *Evolutionary Electronics: Automatic Design of Electronic Circuits and Systems by Genetic Algorithms*. Boca Raton, FL, USA: CRC Press, Inc., 2001.

[4] A. Thompson, P. Layzell, and R. S. Zebulum, "Explorations in design space: Unconventional electronics design through artificial evolution," *IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION*, vol. 3, no. 3, pp. 167–196, 1999.

[5] E. R. Miranda, *Composing music with Computers*. Burlington, MA: Focal Press, 2001.

[6] N. Collins, *Handmade Electronic Music. The art of hardware hacking*. New York–London: Routledge, 2006.

[7] R. Ghazala, *Circuit-Bending. Build your own alien instruments*. Indianapolis: Wiley, 2005.

[8] R. Salminen. (2000) Design Your Own Distortion. [Online]. Available: http://www.generalguitargadgets.com/how-to-build-it/technical-help/articles/design-distortion/

[9] P. Horowitz and W. Hill, *The Art of Electronics*, 3rd ed. New York, NY, USA: Cambridge University Press, 1989.

[10] VV.AA. (2012) DIY Circuit Design: Waveform Clamping. [Online]. Available: https://www.engineersgarage.com/tutorials/diy-circuit-design-waveform-clamping?page=3

[11] T. Q. A.R.Newton, D.O.Pederson, and A.Sangiovanni-Vincentelli, "SPICE3 Version 3f3 User's Manual," Department of Electrical Engineering and Computer Sciences, Berkeley, CA, Usser's manual, 1993.

[12] P. Prusinkiewicz and A. Lindenmayer, *The Algorithmic Beauty of Plants*. New York: Springer, 1990.

[13] J. A. Kaandorp and J. Kübler, *The Algorithmic Beauty of Seaweeds, Sponges, and Corals*. New York, NY, USA: Springer-Verlag New York, Inc., 2001.

[14] H. Meinhardt, *The Algorithmic Beauty of Sea Shells*, 3rd ed. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2003.

[15] E. Gansner, E. Koutsofios, and S. North, *Drawing graphs with dot*, 2006.

[16] S. Wilson, D. Cottle, and N. Collins, Eds., *The SuperCollider Book*. Cambridge, Mass.: The MIT Press, 2011.

[17] G. Brocard, *The LTSpice IV Simulator: Manual, Methods and Applications*. Künzelsau: Swiridoff Verlag, 2011.

CONSTRUCTING PEDB 2nd EDITION: A MUSIC PERFORMANCE DATABASE WITH PHRASE INFORMATION

Mitsuyo Hashida
Soai University
hashida@soai.ac.jp

Eita Nakamura
Kyoto University
enakamura@sap.ist.i.kyoto-u.ac.jp

Haruhiro Katayose
Kwansei Gakuin University
katayose@kwansei.ac.jp

ABSTRACT

Music performance databases that can be referred to as numerical values play important roles in the research of music interpretation, the analysis of expressive performances, automatic transcription, and performance rendering technology. The authors have promoted the creation and public release of the CrestMusePEDB (Performance Expression DataBase), which is a performance expression database of more than two hundred virtuoso piano performances of classical music from the Baroque period through the early twentieth century, including music by Bach, Mozart, Beethoven and Chopin. The CrestMusePEDB has been used by more than fifty research institutions around the world. It has especially contributed to research on performance rendering systems as training data. Responding to the demand to increase the database, we have started a new three-year project to enhance the CrestMusePEDB with a 2nd edition that started in 2016. In the 2nd edition, phrase information that pianists had in mind while playing the performance is included, in addition to the performance data that contain quantitative data. This paper introduces an overview of the ongoing project.

1. INTRODUCTION

The importance of music databases has been recognized through the progress of music information retrieval technologies and benchmarks. Since the year 2000, some large-scale music databases have been created and have had a strong impact on the global research arena [1–4]. Meta-text information, such as the names of composers and musicians, has been attached to large-scale digital databases and has been used in the analysis of music styles, structures, and performance expressions, from the viewpoint of social filtering in MIR fields.

The performance expression data plays an important role in formulating impressions of music [5–8]. Providing a music performance expression database, especially describing deviation information from neutral expression, can be regarded as a research in sound and music community (SMC). In spite of there being many music researches using music performance data, few projects have dealt with creation of a music performance database open to public.

In musicological analysis, some researchers constructed a database of the transition data of pitch and loudness and then use the database through statistical processing. Widmer *et al.* analyzed deviations in the tempi and dynamics of each beat from Horowitz's piano performances [9]. Sapp *et al.*, working on the Mazurka Project [10], collected as many recordings of Chopin mazurka performances as possible in order to analyze deviations of tempo and dynamics by each beat in a similar manner to [9].

The authors have been promoting the creation and public release of the CrestMusePEDB (Performance Expression DataBase), which consists of more than two hundred virtuoso piano performances of classical music from the Baroque period through the early twentieth century, including music by Bach, Mozart, Beethoven, and Chopin [11]. The 1st edition of the CrestMusePEDB (ver.1.0-3.1) has been used musical data and has been used by more than fifty research institutions throughout the world. In particular, it has contributed to researches on the performance rendering systems as training data [12, 13].

The database is unique in providing a set of detailed data of expressive performances, including the local tempo for each beat and the dynamics, onset time deviation, and duration for every note. For example, the performance elements provided in the Mazurka Projects data [10] are beatwise local tempos and dynamics and precise information with note-wise performance elements that cannot be extracted. In the MAPS database [14], which is widely used for polyphonic pitch analysis and piano transcription, performance data does not include any temporal deviations and thus cannot be thought of as realistic performance data in the aspect of musical expression. Such detailed performance expression data are crucial for constructing performance rendering systems and realistic performance models for analysis and transcription.

The size of the CrestMusePEDB 1st edition is not large enough, compared with other databases for computer science. Demand for the database has been increasing in recent years, particularly in the studies using machine learning techniques. In addition, data that explicitly describe the relationship between a performance and the musical structure intended by the performer has been required [1]. Responding to these demands, we started a three-year project in 2016 to enhance the CrestMusePEDB in a 2nd edition,

[1] In many cases, the apex (the most important) note in a phrase is selected by the performer, and there may be the case that phrase sections are analyzed based on the performers own interpretation.

which is described in this paper.

2. CRESTMUSEPEDB 1ST EDITION

The 1st edition of the database CrestMusePEDB [2] aimed to accumulate descriptions of concrete performance expressions (velocity, onset timing, etc.) of individual notes as deviation information from ‘mechanical’ performance. It was focused on classical piano music from the Baroque period through the early twentieth century, including music by Bach, Mozart, Beethoven and Chopin. We chose 51 music pieces, including those often referred to by previous musical studies in the past couple of decades. We also chose various existing recordings by professional musicians. The database contains 121 performances played by one to ten players for each score.

The CrestMusePEDB 1st edition consisted of the following four kinds of component data.

PEDB-SCR (score text information): The score data are included in the database. Files in the MusicXML format and in the standard MIDI file (SMF) format are provided.

PEDB-IDX (audio performance credit): The catalogs of the performances from which expression data are extracted: album title, performer name, song title, CD number, year of publication, etc.

PEDB-DEV (performance deviation data): Outline of curves of tempo and dynamics, the delicate control of each note; deviation regarding starting time, duration, and dynamics from the tactus standard corresponding to the note. Performances from 1 to 10 performers were analyzed for each piece, and multiple deviation data were analyzed by different sound sources and provided for each performance. All data are described in the DeviationInstanceXML format [15, 16].

PEDB-STR (musical structure data): This contains the *estimated* information on a musical structure data set (hierarchical phrase structure and the top note of a phrase) from performances. The data are described in the compliant MusicXML format. The structure data corresponds with a performance expression data in PEDB-DEV. However, if multiple adequate interpretations exist in a piece, the multiple structure data are provided in the performance data.

PEDB-REC (original recordings): The recorded audio performance data were based on the PEDB-STR. Nine players provided 121 performances to compare different expressions from a similar phrase structure.

The primary data were given in SCR (score text information data) and DEV (performance deviation data) files. CrestMusePEDB does not contain any acoustic signal data (WAV files, AIFF files, MP3 files) except for PEDB-REC. Instead, it contained the catalogs of the performances from which expression data were extracted.

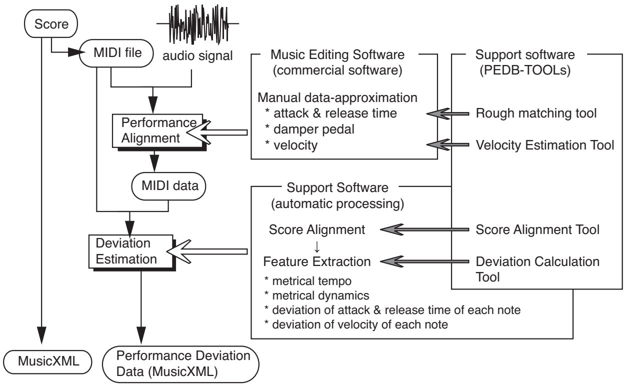

Score
MIDI file
audio signal
Performance Alignment
MIDI data
Deviation Estimation
MusicXML
Performance Deviation Data (MusicXML)
Music Editing Software (commercial software)
Manual data-approximation
* attack & release time
* damper pedal
* velocity
Support software (PEDB-TOOLs)
Rough matching tool
Velocity Estimation Tool
Support Software (automatic processing)
Score Alignment
↓
Feature Extraction
* metrical tempo
* metrical dynamics
* deviation of attack & release time of each note
* deviation of velocity of each note
Score Alignment Tool
Deviation Calculation Tool

Figure 1. Outline of database generation (1st edition)

Transcribing a performance audio signal into MIDI-level data was the core issue for constructing the CrestMusePEDB. Although we had improved the automation of the procedure, an iterated listening process by more than one expert with a commercial audio editor and original alignment software possessed higher reliability.

Fig. 1 illustrates the procedure for generating the PEDB-DEV information.

3. PEDB 2ND EDITION OVERVIEW

The 1st edition of the CrestMusePEDB has contributed to the SMC field, especially for performance rendering studies. The mainstream of the current performance systems refers to the existing performance data. Above all, systems based on recent machine learning techniques require large corpora. The size of the 1st edition CrestMusePEDB is not necessarily large, compared with the other databases published for the research of natural language processing or speech recognition. Demand for the database enhancement has been recently increasing.

Another expectation imposed on the performance database is the handling of information of the musical structure. Although virtuoso performances remain in the form of an acoustic signal, it is hard to find a material that shows the relationship between a performance and its musical structure that the performer intended. In many cases, we had no choice but to estimate the performers intention from the recorded performances.

Responding to these demands, we started a new three-year project in 2016, to enhance the database with a 2nd edition, with the goals of increasing the data size and providing structural information. One of the major problems with making the 1st edition was the workload required for the manual transcription processing of performances in the form of an acoustic signal. To solve this problem, we newly recorded performance data using YAMAHA Disklavier, with the cooperation of skillful pianists who have won prizes in piano competitions. This procedure enabled us to obtain music performance control data (MIDI) and acoustic data simultaneously.

To improve the efficiency of further analysis and utiliza-

[2] http://crestmuse.jp/pedb/

tion of the database, each note in the performance data should be given information of the corresponding note in its musical score. For this goal, the matching file is generated using our original score-performance alignment tool.

We released the beta version of the 2nd edition of the PEDB, which includes data of approximately 100 performances, at the end of May 2017, to meet users requirements regarding data format and data distribution methods. The beta version will include recorded MIDI files, score MIDI files, score files in the MusicXML format, musical structure data, and matching files. Performance in acoustic signals are also included as a reference material. Music structure data released in the beta version include phrase and sub-phrase, and apex notes in each phrase, which are obtained by interviewing the pianists, in a pdf format. Here, the apex note is the most important note from the pianists' perspective in each phrase. We are planning to discuss with the user group of the database the format of the machine-readable structure data and the deviation to be included in the next formal version with the user-group of the database.

4. PROCEDURE FOR MAKING THE DATABASE

4.1 Overview

The main goals of this edition are to enhance the performance data and to provide structural information paired with the performance data. The outline of database generation is shown in Fig. 2.

Musical structure differs depending on pianists' interpretation and even on the score version. Some of the musical works have multiple versions of the musical scores; such as the Paderewski Edition, the Henle Verlag, and the Peters Edition, e.g., Mozart's famous piano sonata in A Major K. 331.

Before the recording, the pianists were asked to prepare to play their most natural interpretation of the score that they usually use. Further they were requested to prepare additional interpretation such as by a different score edition, by a professional conductor suggests, and by over-expressed the pianists' interpretation. Pianists were requested to express the difference of these multiple interpretations regarding the phrase structure. After the recording, we interviewed the pianist on how (s)he tried to express the intended structure of the piece after listening to the recorded performances.

Through these steps, source materials for the database are obtained. Then, the materials are analyzed and converted to the data, as they can be referenced in the database. The main procedure of this stage is note-to-note matching between score data and the performance. Some notes in the performance may be erroneously played, and the number of notes played in trill is not constant. To handle such situations, we developed 'an original score-performance alignment tool based on a hidden Markov model (HMM) [17].

In the following subsections, the recording procedure and the overview of the alignment tool are described.

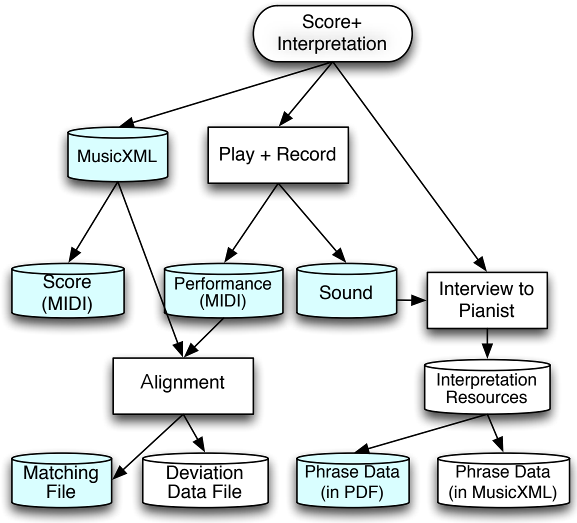


Figure 2. Outline of the database generation. Blue: data included in the beta version.

4.2 Recording

The key feature in the creation of the 2nd edition PEDB is that we can talk with the pianists directly about their performances. Before the recording procedure, we confirmed with each pianist that the recorded performance should clarify its phrase structure (phrase or sub-phrase) and its apex note, as the "interpretation" of the performance.

Pianists are asked to (1) play based on their own interpretation for all pieces and to (2) play with exaggerated expressions retaining the phrase structure for some pieces. In addition, for some pieces, pianists are asked to (3) play with the intension of accompanying a soloist or dancers. If there are different interpretations or score editions of one piece, they played with the both versions. For Mozart's piano sonata in A Major K. 331 and Beethoven's "Pathetique" Sonata, the 2nd movement, different versions of the scores have been provided to the pianists by the authors.

Recordings were done in several music laboratories or recording studios. As shown in Fig. 3, performances played with a YAMAHA Disklavier were recorded as both audio signals and MIDI data including controls of pedals, via ProTools. We also recorded the video for the interview process after recording.

4.3 Alignment Tool

After MIDI recordings are obtained, each MIDI signal is aligned to the corresponding musical score. To improve the efficiency, this is done in two stages: automatic alignment and correction by an annotator. In the automatic alignment stage, a MIDI performance is analyzed with an algorithm based on an HMM [17], which is one of the most accurate and fast alignment methods for symbolic music. A post-processing to detect performance errors, i.e., pitch errors, extra notes, and missing notes, is also included in this stage.

In the following stage of correction by an annotator, a vi-

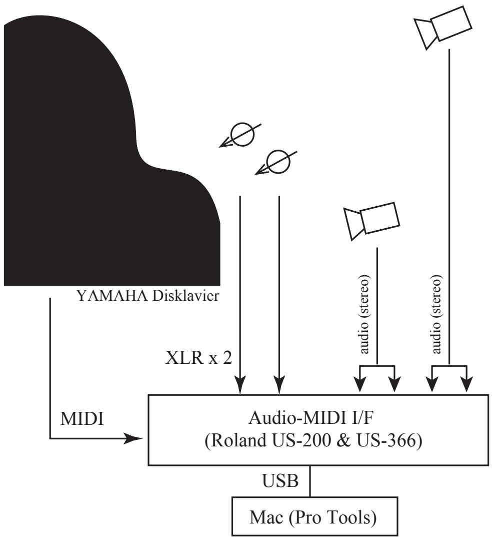


Figure 3. Technical setup for the recording

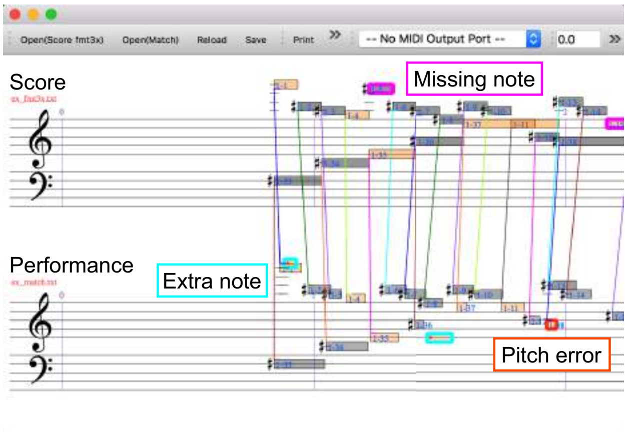


Figure 4. A visualization tool used to obtain the alignment information (see text).

sualization tool called *Alignment Editor* is used to facilitate the procedure. In the editor, the performance and score are represented as piano rolls, and the alignment information is represented as lines between the corresponding score notes and performed notes, as well as indications for performance errors (Fig. 4). On each illustrated note, an ID referring to the score notes is also presented, and the annotator can edit it to correct for the alignment. Upon saving the alignment information, the editor automatically reruns the performance error detection and updates the graphic.

5. PUBLISHING THE DATABASE

We released a beta version of the 2nd edition PEDB consisting of 103 performances of 43 pieces by two professional pianists at the end of May, 2017. Table 1 shows the list of the performances included in the beta version. As shown in this table, some of the pieces are played with more than one expression or interpretation.

This edition provides (1) recording files (WAV and MIDI), (2) score files (MusicXML and MIDI), (3) information regarding phrase and apex notes in phrases by in PDFs (Fig. 5) and (4) the alignment information by the original file format (*matching file* format) (see Fig. 2.)

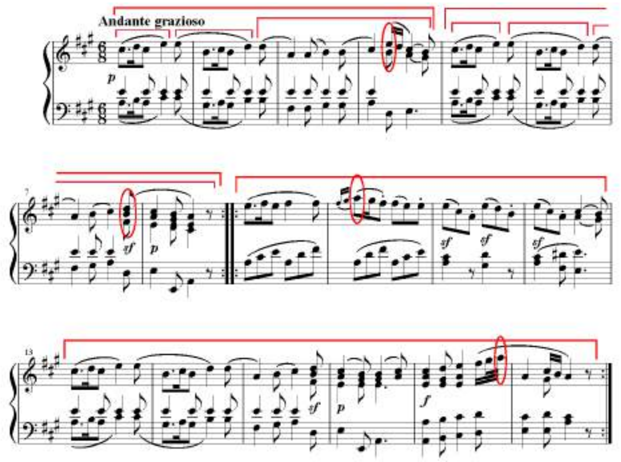


Figure 5. A sample of phrase structure data (W. A. Mozart's Piano Sonata K. 331, 1st Mov., Peters Edition.) Square bracket and circle mark denotes phrase/sub-phrase and apex, respectively.

In a matching file format, the recorded MIDI performance is represented as a piano roll. For each note, onset time, offset (key-release) time, pitch, onset velocity, and offset velocity are extracted and presented. In addition, the corresponding score note, the score time, and performance error information are provided for each performed note. To represent the performance error information, each performed note is categorized as a correct note, a pitch error, or an extra note, according to the results of the score-to-performance alignment. In the case of an extra note, no corresponding score note is given. The file also describes a list of missing score notes that have no corresponding notes in the performance.

As shown in Fig. 4, the durations of the performed notes are usually played shorter where the damper pedal is pressed. When pressed, the damper pedal sustains the sound of the notes until the pedal is released. It means that there are two interpretations for note offset (key-release) time: actual note-off time and pedal-release time. In a matching file format, we adopted the actual note-off time, as a offset (key-release) time. The pedal information is included in the recorded MIDI files.

The database is available from the PEDB 2nd Edition url [3] .

6. CONCLUDING REMARKS

In this paper, we introduced our latest attempt to enhance the PEDB and overviewed part of the database as a beta version to investigate the users requests for the database. Although the number of data currently collected in the beta version is small, we have completed the workflow for the database creation. In the coming years, we plan to increase the number of performance data to more than five hundred

[3] http//:crestmuse.jp/pedb_edition2/

Table 1. List of performances included in the beta version. *self*: the player's expression, *over*: over expression, *accomp.*: played as the accompaniment for solo instrument or chorus, *waltz*: focused on leading a dance, *Hoshina*: played along with a professional conductor's interpretation, *Henle* and *Peters*: used the score of Henle Edition and Peters Edition, and others: extra expressions via discussion with the authors and each player.

No.	Composer	Title	Performances			
			Player 1		Player 2	
			#	interpretation	#	interpretation
1	J. S. Bach	Invention No. 1	2	self / over	2	self / over
2	J. S. Bach	Invention No. 2	2	self / over	1	self
3	J. S. Bach	Invention No. 8	-	-	1	self
4	J. S. Bach	Invention No. 15	2	self / over	1	self
5	J. S. Bach	Wohltemperierte Klavier I-1, prelude	2	self / accomp.	-	
6	L. V. Beethoven	Für Elise	1	self	3	self / over / note_d
8	L. V. Beethoven	Piano Sonata No. 8 Mov. 1	1	self	1	self
9	L. V. Beethoven	Piano Sonata No. 8 Mov. 2	2	self / Hoshina	2	self / Hoshina
10	L. V. Beethoven	Piano Sonata No. 8 Mov. 3	1	self	1	self
7	L. V. Beethoven	Piano Sonata No. 14 Mov. 1	2	self / over	1	self
11	F. Chopin	Etude No. 3	2	self / over	1	self
12	F. Chopin	Fantaisie-Impromptu, Op. 66	2	self / over	1	self
15	F. Chopin	Mazurka No. 5	1	self	-	
13	F. Chopin	Mazurka No. 13	2	self / over	-	
14	F. Chopin	Mazurka No. 19	2	self / over	-	
16	F. Chopin	Nocturne No. 2	2	self / over	1	self
17	F. Chopin	Prelude No. 1	2	self / over	-	
18	F. Chopin	Prelude No. 4	1	self	1	self
19	F. Chopin	Prelude No. 7	1	self	1	self
20	F. Chopin	Prelude No. 15	1	self	1	self
21	F. Chopin	Prelude No. 20	1	self	-	
22	F. Chopin	Waltz No. 1	2	self / waltz	2	self / waltz
23	F. Chopin	Waltz No. 3	2	self / over	1	self
24	F. Chopin	Waltz No. 7	2	self / waltz	2	self / waltz
25	F. Chopin	Waltz No. 9	2	self / over	1	self
26	F. Chopin	Waltz No. 10	1	self	1	self
27	C. Debussy	La fille aux cheveux de lin	2	self / over	-	
28	C. Debussy	Rêverie	1	self	-	
29	E. Elgar	Salut d'amour Op. 12	2	self / accomp.	-	
30	G. Händel	Largo / Ombra mai fù	2	self / accomp.	-	
31	Japanese folk song	Makiba-no-asa	2	exp1 / exp2	-	
32	F. Liszt	Liebesträume	1	self	-	
33	F. Monpou	Impresiones intimas No. 5 "Pajaro triste"	1	self	-	
34	W. A. Mozart	Piano Sonata K. 331 Mov. 1	3	self / Henle / Peters	2	Henle / Peters
35	W. A. Mozart	Twelve Variations on "Ah vous dirai-je, Maman"	2	self / over	-	
36	T. Okano	Furusato	2	self / accomp.	2	self / accomp
37	T. Okano	Oboro-zuki-yo	2	self / accomp.	-	
38	S. Rachmaninov	Prelude Op. 3, No. 2	2	self / neutral	-	
39	M. Ravel	Pavane pour une infante défunte	1	self	-	
40	E. Satie	Gymnopédies No. 2	2	self / neutral	-	
41	R. Schumann	Kinderszenen No. 7 "Träumerei"	2	self / neutral	1	self
42	P. I. Tchaikovsky	The Seasons Op. 37b No. 6 "June: Barcarolle"	2	self / over	-	
		Total:	70		31	

by adding new data to the previous version, considering the compatibility with the data-format of the PEDB 1st edition.

There is no other machine-readable performance database associated with musical structure. We hope that the database can be utilized for research in many research fields related to music performances. As future work, we would like to develop some applications, in addition to the data-format design, so the database can be used by researchers who are not familiar with information technology.

Acknowledgments

The authors are grateful to Dr. Y. Ogawa and Dr. S. Furuya for their helpful suggestion. We also thank Ms. E. Furuya for her advice as a professional pianist, and Professor T. Murao for his cooperation. This work was supported by JSPS KAKENHI Grant Numbers 16H02917, 15K16054, and 16J05486. E. Nakamura is supported by the JSPS research fellowship (PD).

7. REFERENCES

[1] J. S. Downie, "The music information retrieval evaluation exchange (mirex)," in *D-Lib Magazine*, 2006, p. Vol.12 No.12.

[2] "http://staff.aist.go.jp/m.goto/rwc-mdb/," 2012 (last update).

[3] D. McEnnis, C. McKay, and I. Fujinaga, "Overview of omen," in *Proc. ISMIR*, Victoria, 2006, pp. 7–12.

[4] H. Schaffrath, "http://essen.themefinder.org/," 2000 (last update).

[5] M. Senju and K. Ohgushi, "How are the player's ideas conveyed to the audience?" in *Music Perception*, vol. 4, no. 4, 1987, pp. 311–323.

[6] H. Hoshina, *The Approach toward a Live Musical Expression: A Method of Performance Interpretation considered with energy*. Ongaku-no-tomo-sha, 1998, (written in Japanese).

[7] N. Kenyon, *Simon Rattle: From Birmingham to Berlin*. Faber & Faber, 2003.

[8] K. Stockhausen, *Texte zur elektronischen und instrumentalen Musik*, J. Shimizu, Ed. Gendai-shicho-shinsha, 1999, (Japanese translation edition).

[9] G. Widmer, S. Dixson, S. Goebl, E. Pampalk, and A. Tobudic, "In research of the Horowitz factor," *AI Magazine*, vol. 24, no. 3, pp. 111–130, 2003. [Online]. Available: https://www.aaai.org/ojs/index.php/aimagazine/article/view/1722/1620

[10] C. Sapp, "Comparative analysis of multiple musical performances," in *Proc. ISMIR*, Vienna, 2007, pp. 497–500.

[11] M. Hashida, T. Matsui, and H. Katayose, "A new music database describing deviation information of performance expressions," in *Proc. ISMIR*, Kobe, 2008, pp. 489–494.

[12] M. Hashida, K. Hirata, and H. Katayose, "Rencon Workshop 2011 (SMC-Rencon): Performance rendering contest for computer systems," in *Proc. SMC*, Padova, 2011.

[13] H. Katayose, M. Hashida, G. De.Poli, and K. Hirata, "On evaluating systems for generating expressive music performance: the rencon experience," in *J. New Music Research*, vol. 41, no. 4, 2012, pp. 299–310.

[14] V. Emiya, R. Badeau, and B. David, "Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle," in *IEEE TASLP*, vol. 18, no. 6, 2010, pp. 1643–1654.

[15] T. Kitahara, "Mid-level representations of musical audio signals for music information retrieval," in *Advances in Music In-formation Retrieval, Springer*, vol. 274, 2010, pp. 65–91.

[16] T. Kitahara and H. Katayose, "CrestMuse toolkit: A java-based frame-work for signal and symbolic music processing," in *Proc. Signal Processing (ICSP)*, Beijing, 2014.

[17] E. Nakamura, N. Ono, S. Sagayama, and K. Watanabe, "A stochastic temporal model of polyphonic midi performance with ornaments," in *J. New Music Research*, vol. 44, no. 4, 2015, pp. 287–304.

MUSICAL FEATURE AND NOVELTY CURVE CHARACTERIZATIONS AS PREDICTORS OF SEGMENTATION ACCURACY

Martin Hartmann
Finnish Centre for
Interdisciplinary Music Research
University of Jyväskylä, Finland
martin.hartmann@jyu.fi

Olivier Lartillot
Department of Musicology
University of Oslo, Norway
olartillot@gmail.com

Petri Toiviainen
Finnish Centre for
Interdisciplinary Music Research
University of Jyväskylä, Finland
petri.toiviainen@jyu.fi

ABSTRACT

Novelty detection is a well-established method for analyzing the structure of music based on acoustic descriptors. Work on novelty-based segmentation prediction has mainly concentrated on enhancement of features and similarity matrices, novelty kernel computation and peak detection. Less attention, however, has been paid to characteristics of musical features and novelty curves, and their contribution to segmentation accuracy. This is particularly important as it can help unearth acoustic cues prompting perceptual segmentation and find new determinants of segmentation model performance. This study focused on spectral, rhythmic and harmonic prediction of perceptual segmentation density, which was obtained for six musical examples from 18 musician listeners via an annotation task. The proposed approach involved comparisons between perceptual segment density and novelty curves; in particular, we investigated possible predictors of segmentation accuracy based on musical features and novelty curves. For pitch and rhythm, we found positive correlates between segmentation accuracy and both local variability of musical features and mean distance between subsequent local maxima of novelty curves. According to the results, segmentation accuracy increases for stimuli with milder local changes and fewer novelty peaks. Implications regarding prediction of listeners' segmentation are discussed in the light of theoretical postulates of perceptual organization.

1. INTRODUCTION

Musical segments are a representation of the perceived structure of music, and hence carry its multifaceted, interwoven, and hierarchically organized nature. Transitional points or regions between segments, which are called *perceptual segment boundaries*, can emerge from temporal changes in one or more musical attributes, or from more complex configurations involving e.g. repetition and cadences. Exhibited acoustic change can be measured to yield an estimate of *novelty* with respect to previous and upcoming musical events, with the aim to delineate perceptual segment boundaries: for each instant, its degree of acoustic novelty is expected to predict the likelihood of indicating a boundary. Segmentation accuracies for different musical features, such as timbre and harmony descriptors, can be obtained via Novelty detection (Foote, 2000) approaches. The success of predicting musical segment boundaries from acoustic estimates of novelty depends on the level of structural complexity of a musical piece. For instance, music that unambiguously evokes few sharp segment boundaries and clear continuity within segments to listeners should exhibit high accuracy, whereas pieces that prompt many boundaries and a rather heterogeneous profile might be more challenging for prediction. Music is however multidimensional, which implies that accuracy for a given stimulus could depend on the musical feature or features under study.

This work focuses on the assessment of possible predictors of segmentation accuracy that could be obtained directly from musical features or from novelty points derived from these features. We explored musical features of different types, because listeners rarely focus on a single dimension of music. To illustrate possible applications of the proposed approach, one could imagine a segmentation system that could extract candidate features from a particular musical piece (Peiszer, Lidy, & Rauber, 2008), and discard any features that would not seem to be informative of musical changes in order to focus only on those changes that would be deemed relevant for a listener. This would result in more efficient prediction, as the system would not require to compute subsequent structure analysis steps for irrelevant features. Our endeavor is motivated by the direct impact of music segmentation on other areas of computational music analysis, including music summarization, chorus detection, and music transcription, and its relevance for the study of human perception, as it can deepen our understanding on how listeners parse temporally unfolding processes.

MIR (Music Information Retrieval) studies on segmentation and structure analysis typically require perceptual data for algorithm evaluation. Often, data collection involves the indication of segment boundaries from one or few listeners for a large amount of musical examples; this results in a set of time points for each piece. In contrast, approaches on music segmentation within the field of music perception and cognition commonly involve the collection of perceptual boundaries from multiple participants in listening experiments; the collected data is often aggregated across listeners for its analysis. Kernel Density Estimation (*KDE*, Silverman, 1986) has been used in recent segmentation studies (e.g. Bruderer, 2008; Hartmann, Lartillot, & Toiviainen, 2016b) to obtain a precise aggregation across boundary data. This approach consists of obtaining a probability density estimate of the boundary data using a Gaussian function; in a KDE curve, temporal regions that prompted boundary indication from many listeners are represented as peaks of perceptual boundary density. This continuous representation of perceptual segment boundary probability has been used to compare different stimuli, groups of participants, and segmentation tasks (Bruderer, 2008; Hartmann et al., 2016b).

Prediction of perceptual boundary density in the audio domain often involves the computation of novelty curves (Foote, 2000), which roughly describe the extent to which a temporal context is characterized by two continuous segments separated by a discontinuity with respect to a given musical feature. Recent studies have compared novelty curves with perceptual boundary density curves (Hartmann, Lartillot, & Toiviainen, 2016a) and compared peaks derived from both curves (Mendoza Garay, 2014), showing that novelty detection can predict segmentation probabilities derived from numerous participants.

A preliminary step in novelty detection and other segmentation frameworks consists of the extraction of a musical feature, which will determine the type of musical contrast to be detected. Timbre, tonality, and to some extent rhythm (Jensen, 2007) have been considered to be important features for structural analysis. Relatively high prediction of musical structure has been found for two musical features: MFCCs (Mel-Frequency Cepstral Coefficients) for timbre description (Foote, 2000) and Chromagram or similar features (Serrà, Muller, Grosche, & Arcos, 2014) for description of pitch changes; also combined approaches have been proposed (Eronen, 2007). The segmentation accuracy achieved by novelty curves seems to highly depend on the musical feature that is used, and on the choice of temporal parameters for feature extraction (Peeters, 2004). In addition, different musical pieces might require different musical features for optimal prediction (Peiszer et al., 2008); as pointed out by McFee and Ellis (2014), structure in pop and rock is frequently determined by harmonic change, whereas jazz is often sectioned based on instrumentation. No single feature can optimally predict all musical examples; certain features are more appropriate than others depending on particular aspects of musical pieces (Smith & Chew, 2013).

This mechanism is however not well understood at present: it is unclear what characteristics of musical features contribute to the segmentation accuracy for a given musical piece. Addressing this issue would be important since it would enable the possibility to select optimal musical features for further novelty detection, avoiding the computation of novelty curves that would not yield satisfactory results; it would also help to develop better alternatives to the novelty detection approach, with the aim of reducing computational costs.

To analyze the impact of different factors associated to novelty curves upon segmentation and compare different segmentation algorithms, a number of performance measures have been proposed, such as precision, recall, and F-measure (Lukashevich, 2008); also correlation between time series has been applied for this purpose (Hartmann et al., 2016a). One of the factors that has been shown to highly contribute to the segmentation accuracy is the width of the novelty kernel (e.g. Hartmann et al., 2016a), which roughly refers to the temporal context with respect to which novelty is estimated for each time point.

However, to the best of our knowledge, no study has systematically investigated what specific aspects of novelty curves contribute to their accuracy for a given stimulus. It would be relevant to investigate what characteristics of novelty curves relate to their accuracy, as this could allow to predict the relative suitability of a novelty curve for a given stimulus without the need of direct comparison against ground truth, and to bypass computation of novelty curves that would be assumed not to deliver satisfactory performance with regard to a particular stimulus. From the viewpoint of music perception, it would be useful to better understand the extent to which musical characteristics perceived by listeners are directly apparent from novelty curves, and to gain more knowledge on the types of musical changes that prompt both boundary perception and high novelty scores.

Recently, Hartmann et al. (2016a) studied segmentation accuracy achieved for concatenated musical pieces using different novelty curves. It was found that optimal prediction of perceptual segment boundary density involves the use of large kernel widths; the study also highlights the role of rhythmic and pitch-based features on segmentation prediction. This study is a follow-up to the paper by Hartmann et al. (2016a), as it focused further on prediction of perceptual segmentation density via novelty detection, and examined the same musical pieces; in particular, we investigated one of the perceptual segmentation sets (an *annotation* segmentation task performed by musician listeners) studied by Hartmann et al. (2016a), and explored perceptual segmentation density and novelty curves for individual musical stimuli. The aim of the present study was to understand whether or not local variability of musical features and distance between novelty peaks are related with the accuracy of segmentation models. The following research questions guided our investigation:

1. What specific aspects of musical stimuli that account for segmentation accuracy can be directly described from musical features?

2. What stimulus-specific attributes of novelty curves determine optimal segmentation accuracy?

As regards the first research question, we expected to find an inverse relationship, dependent on musical stimulus, between magnitude of local feature variation and accuracy obtained via novelty detection. For instance, musical stimuli displaying unfrequent local tonal contrast would yield optimal segmentation accuracy via tonal novelty curves. The rationale behind this hypothesis is that if there is not much local change in a feature, then the local changes that occur in that feature should be more salient. One of the most basic

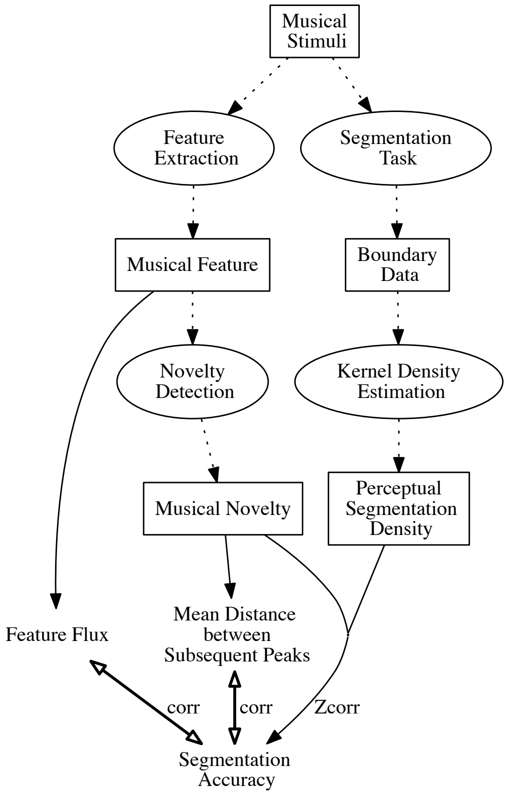


Figure 1. General design of the study.

Gestalt principles, the law of *Prägnanz*, relates to this rationale, because it states that, under given conditions, perceptual organization will be as "good" as possible, i.e. percepts tend to have the simplest, most stable and most regular organization that fits the sensory pattern (Koffka, 1935).

Regarding the second question, we expected that stimuli yielding higher accuracy for a given feature would exhibit a relatively large temporal distance between novelty peaks for that feature. Although this relationship could depend on other factors such as tempo and duration, we believed that the absolute time span between peaks would serve as a rough predictor of accuracy for the reasons above mentioned regarding the law of Prägnanz. To give an example, optimal accuracy was expected for stimuli characterized by long and uniform segments with respect to instrumentation that would be delimited by important timbral changes, whereas music characterized by more frequent timbral change and gradual transitions would yield lower accuracy for a timbre-based novelty measure.

2. METHOD

Figure 1 illustrates the research design utilized in our investigation. The upper part of the figure concerns computational prediction of segmentation via novelty detection and a perceptual modelling of collected segmentation data via perceptual segmentation density estimation. These two topics were more thoroughly covered in Hartmann et al. (2016b) and Hartmann et al. (2016a), respectively. The bottom part of the figure, specifically the solid line connections, refer to investigation of correlates of perceptual boundary density prediction, which is the main focus of this study.

2.1 Segmentation Task

To obtain perceptual segment boundary density of the stimuli, we collected boundary data from participants via a listening task that involved an offline annotation. This non real-time segmentation task, called *annotation task*, is described with more detail in Hartmann et al. (2016b). The reason for analyzing prediction of non real-time segmentation was to reduce the number of intervening factors: compared to a real-time segmentation, boundary placements obtained via offline segmentation are probably better aligned in time with respect to musical changes; also, inter-subject agreement has been found to be higher for offline than for online tasks (Hartmann et al., 2016b).

2.1.1 Subjects

18 musicians (11 males, 7 females) with a mean age of 27.61 years (*SD* = 4.45) and an average musical training of 14.39 years (*SD* = 7.49) took part of this experiment. They played classical (12 participants) and non-classical musical styles (6 participants) such as rock, pop, folk, funk, and jazz.

2.1.2 Stimuli

We selected 6 instrumental music pieces; two of them lasted 2 minutes and the other four were trimmed down to this length for a total experiment duration of around one hour. The pieces (see Hartmann et al., 2016a) comprise a variety of styles and considerably differ from one another in terms of musical form; further, they emphasize aspects of musical change of varying nature and complexity.

2.1.3 Apparatus

An interface in Sonic Visualizer (Cannam, Landone, & Sandler, 2010) was prepared to obtain segmentation boundary indications from participants. Stimuli were presented in randomized order; for each stimulus, the interface showed its waveform as a visual-spatial cue over which boundaries would be positioned (subjects were asked to focus solely on the music). Participants used headphones to play back the music at a comfortable listening level, and both keyboard and mouse were required to complete the segmentation task.

2.1.4 Procedure

Written instructions were given to participants; these included a presentation of the interface tools and a task description, which consisted of the following steps:

1. Listen to the whole musical example.

2. Indicate significant instants of change while listening to the music by pressing the Enter key of the computer.

3. Freely play back from different parts of the musical example and make the segmentation more precise by adjusting the position of boundaries; also removal of any boundaries indicated by mistake is allowed.

4. Rate the perceived strength of each boundary (ratings of boundary strength were collected for another study). Start over from the first step for the next musical example.

2.2 Perceptual Segment Boundary Density

We obtained a perceptual boundary density estimate across the segmentation data collected from musician participants; this estimate would be further compared against novelty curves to assess their accuracy. The perceptual boundary data of all participants was used to obtain a curve of perceptual segmentation density using a KDE bandwidth of 1.5 s; values around this bandwidth were found optimal for comparison between perceptual segmentation densities (Hartmann et al., 2016b). From each tail of the perceptual density curves, 6.4 s were trimmed for more accurate comparisons with novelty curves (see below).

2.3 Feature Extraction and Novelty Detection

We computed novelty curves from 5 musical features describing timbre (Subband Flux), rhythm (Fluctuation Patterns), pitch class (Chromagram) and tonality (Key Strength, Tonal Centroid) using MIRtoolbox 1.6.1 (Lartillot & Toiviainen, 2007); see Hartmann et al. (2016a) for a description of the features used for novelty detection. For each feature, a self-similarity matrix was obtained by computing the cosine distance between all possible pairs of feature frames. Novelty for each time point was computed via convolution between each self-similarity matrix and a Gaussian checkerboard kernel (Foote, 2000) with half width of 11 s; large kernel sizes have been previously used to overcome high levels of detail in novelty curves (e.g. Hartmann et al., 2016a).

As done with the perceptual density curves, we truncated the novelty curves by trimming 6.4 s from each extreme to avoid edge effects. We chose the smallest value that would eliminate, for all stimuli, any novelty spikes caused by the contrast between music and silence in the beginning and end of tracks; trimming the extremes of the novelty curves also increased the number of novelty points that derive from a full checkerboard kernel. Once the novelty curves were computed, we also trimmed 6.4 s from the extremes of each dimension of the musical features in order to obtain the predictors of accuracy described below.

2.4 Characterization of Musical Features and Novelty Curves

Subsequently, we computed two characterizations in order to estimate feature local discontinuity and temporal distance between music structure changes. From each musical feature matrix F we calculated mean *Feature Flux*, an estimate of the amount of local variation; Feature Flux is the Euclidean distance between successive feature frames. First, for each time series F_d, where d corresponds to a feature dimension, the squared difference between successive time points is obtained. Next, a flux time series v is obtained as the squared root of the sum across dimensions:

$$v_t = \sqrt{\sum_{d=1}^{N}(F_d(t) - F_d(t-1))^2}$$

Finally, mean Feature Flux is obtained by averaging the flux time series v across time points:

$$\mathit{Feature\ Flux} = \frac{1}{K}\sum_{t=1}^{K} v_t$$

From each novelty curve, we obtained *Mean Distance Between Subsequent Peaks* (MDSP), which describes the peak-to-peak duration (in seconds) of novelty curves. To compute this estimate, we first obtained from the novelty curve a vector of novelty peak locations v, where v_i corresponds to the ith peak, and N corresponds to the number of peaks; MDSP was calculated as follows:

$$\mathit{MDSP} = \frac{1}{N}\sum_{i=1}^{N}(v_i - v_{i-1})$$

2.5 Segmentation Accuracy and its Correlates

We compared perceptual segmentation density and novelty curves to obtain segmentation accuracy. To this end, we performed correlations between novelty and perceptual segmentation density for each stimulus and musical feature.

We focused on the possible relationship between accuracy and the aforementioned characterizations of musical features and novelty peaks. Hence, for each feature we correlated across stimuli the accuracy with Feature Flux and with MDSP. In order to perform these correlations, the accuracies of each feature required to follow an approximately normal distribution, so we subsequently transformed accuracies via Fisher's z transformation of r. The normalization of Z_r involved the calculation of effective degrees of freedom to correct for temporal autocorrelation (Pyper & Peterman, 1998).

3. RESULTS

3.1 Prediction of Perceptual Segmentation Density

Figure 2 shows the correlation between novelty curves and perceptual boundary density for each stimulus. The prediction accuracies were found to vary depending on stimulus; for instance, *Smetana* yielded very high correlations, whereas these were rather low for *Couperin*. Also, for any given stimulus, accuracy differed according to the feature and feature type used for novelty detection; for instance, *Smetana* yielded higher accuracy for pitch-based features than for rhythmic and spectral-based features. Interestingly, no single novelty feature successfully predicted perceptual boundary density for all stimuli.

At this point, it might be relevant to illustrate the data analyzed in this study and at the same time explore two

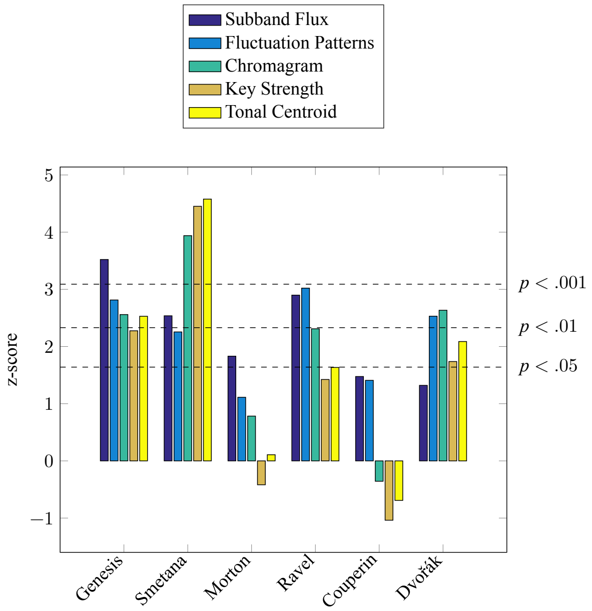


Figure 2. Correlation between novelty curves and perceptual boundary density.

musical stimuli that were found to be contrasting with respect to their prediction. Figure 3 visually compares representations for two stimuli that yielded optimal and worst accuracies for Tonal Centroid, *Smetana* and *Couperin*, respectively. The upper graphs show the Tonal Centroid time series for each stimulus; dimensions 1-2, 3-4 and 5-6 correspond to pairs of numerical coordinates for the circle of fifths, minor thirds, and major thirds, respectively. The middle graphs present novelty curves for Tonal Centroid, which result from the representations in the upper graphs. The lower graph shows perceptual segmentation density based on listeners' segmentation. Some clear differences can be seen between the profiles for *Smetana* and *Couperin*. The three most contrasting peaks of perceptual density for *Smetana* correspond to changes of key: at 40 s the music changes from an F key to a fugue in G min, at 66 s the F melody is reprised in C♯, and at 104 s begins a similar fugue transposed to F min. These changes are quite clear in the Tonal Centroid representation, yielding a relatively adequate novelty prediction, although the rather gradual transition at around 66 s does not yield a stark novelty peak.

In contrast, the Tonal Centroid time series of *Couperin* does not allow for a straightforward acoustic interpretation of the perceptual segmentation density profile. According to the peaks of perceptual density, listeners seem to base their indications primarily on the ending of cadences; these are characterized by the use of heavy ornamentation (e.g. mordents) and chords, which are rather salient because the piece is almost exclusively two-voiced. Due to these cues, listeners seem to place more segment boundaries on endings of melodies than on beginnings, for instance at 23 s. In comparison to this, Tonal Centroid and its corresponding novelty curve would describe slightly delayed musical changes, as harmonic transitions become apparent only after enough development of subsequent melodic material. Another reason behind the highly inaccurate prediction is that listeners placed boundaries for changes of register, rests and durational changes, which might partly explain the higher accuracies obtained for spectral and rhythmic features.

3.2 Finding Predictors of Segmentation Accuracy

Musical Feature	Feature Flux	MDSP
Subband Flux	.54	–.03
Fluctuation Patterns	–.30	.11
Chromagram	–.68	.27
Key Strength	**–.74**	.47
Tonal Centroid	–.66	**.50**

Table 1. Correlation between z-transformed accuracy and characterizations of musical features (Feature Flux) and novelty curves (MDSP).

Subsequently, we analyzed characterizations of extracted features and of novelty curves derived therefrom, looking for correlates of accuracy. We focused on Feature Flux, a global estimate of the extracted features, and on MDSP, which was obtained from novelty curves, to find whether or not these would be indicative of novelty curve accuracy. Table 1 shows the correlation between segmentation accuracies and the obtained characterizations of musical features and novelty curves. We found a strong negative correlation between accuracy and Feature Flux for pitch-based features; as regards accuracy and MDSP, we obtained moderate to strong positive correlations for tonal features (*strong* and *moderate* mean $|r| > .5$ and $.3 < |r| < .5$ respectively, following Cohen, 1988). Although these results did not reach statistical significance at $p < .05$, some interpretations can still be made. According to the results, accuracy increases for stimuli with fewer local change in pitch content and less peaks in pitch-based novelty curves. A similar pattern of results was found for rhythm; we obtained for Fluctuation Patterns a moderate negative correlation between Feature Flux and accuracy, and a weak positive correlation between MDSP and accuracy. Timbre seemed to yield an opposite trend, at least for Feature Flux; Subband Flux exhibited a strong positive correlation between Feature Flux and accuracy, and no or very weak correlation between MDSP and accuracy. This suggests that high accuracy is associated with more local variability of spectral fluctuation.

4. DISCUSSION

Understanding which specific aspects of musical pieces influence novelty-based segmentation prediction is a crucial but challenging issue. One possible way to address this problem is to focus on the particulars of this approach and tackle the question of what characteristics of musical features and their respective novelty curves predict segmentation accuracy for different musical pieces. This study tries to fill the gap in this respect, and aims to open a discussion on the possibility of predicting accuracy directly from

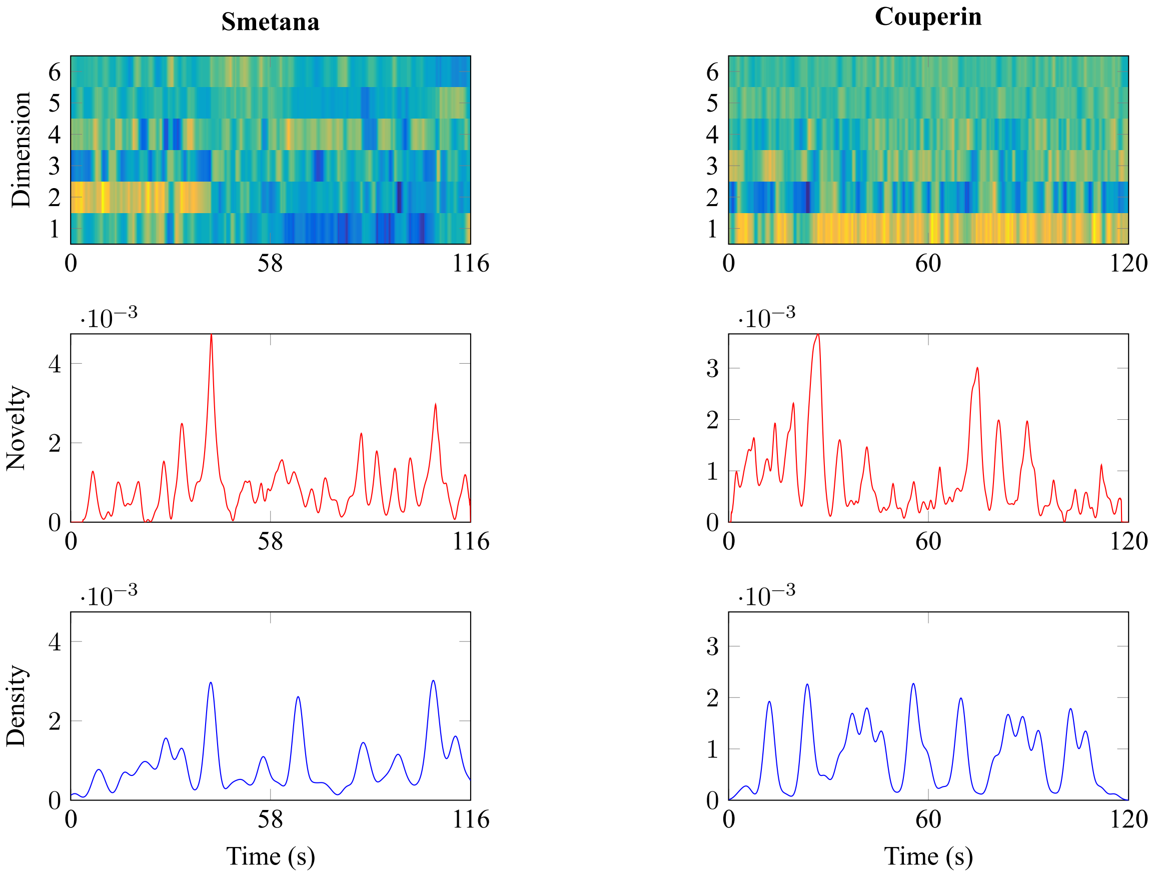


Figure 3. Tonal Centroid, novelty of Tonal Centroid and perceptual boundary density for stimuli *Smetana* and *Couperin*.

musical feature characteristics; we believe that automatic segmentation systems could optimize their parameters for stimuli of different idioms and that such an approach would lead to an increase in accuracy.

Regarding the validity of the approach, we also highlight that our method for studying segmentation involves amassing annotations from multiple listeners for the same stimulus; this possibility has been previously explored only by few studies on segmentation prediction (Mendoza Garay, 2014; Hartmann et al., 2016a). In contrast, MIR studies on music segmentation are often based on data coming from one or few annotators; the computation of a KDE is usually not needed since the set of annotated segment boundaries is directly compared against peaks picked from a novelty curve. In this sense, analyses of perceptual segmentation based on data that is probably more representative of the musician population should be useful for better understanding both perception and prediction of musical structure.

This section examines our research questions in light of the proposed analysis, and assesses the extent to which the stated hypotheses could be supported. Finally, we conclude this article with possible directions for future research.

4.1 Segmentation Accuracy

The first step to address the research questions was to investigate the accuracy to predict perceived boundaries yielded by different musical features for different musical examples. As shown in Figure 2, accuracy seems to highly vary according to musical piece, motivating further analyses on novelty detection that focus on each piece separately. It is also apparent that no single algorithm is robust for prediction of all examples, suggesting the importance of incorporating combinations of multiple features (Hartmann et al., 2016a; Müller, Chew, & Bello, 2016), multiple time scales (Kaiser & Peeters, 2013; Gaudefroy, Papadopoulos, & Kowalski, 2015), and other aspects of segmentation (e.g. repetition principles, see Gaudefroy et al., 2015; Paulus & Klapuri, 2006) into novelty approaches. Overall, however, the results seem to support the idea that performance in structural analysis heavily may depend more on musical stimuli than on the algorithm used or the choice of parameters (Peiszer et al., 2008), which could serve to justify subsequent steps of our analysis.

4.2 Feature-based Prediction of Novelty Detection Performance

Our first hypothesis was that accuracy obtained via novelty detection would increase for stimuli with low local variation of musical features. In support with this hypothesis, we overall found Feature Flux to be a good predictor of correlation between perceptual segmentation density and novelty (Table 1), and a same pattern of results for pitch-based and rhythmic features. This suggests, for these features, that increased local feature continuity may be indicative of higher accuracy of novelty curves.

The second hypothesis of this study was that accuracy would increase for stimuli with more distant novelty peaks. Indeed, mean distance between subsequent peaks was found to be somewhat indicative of the accuracy of novelty curves with respect to perceptual segmentation density. This suggests, particularly for tonal features, that longer novelty peak-to-peak duration is associated with higher correlation between detected novelty and perceptual segment density.

Focusing on pitch-based and rhythmic features, the results indicate that high local variability is associated with low accuracy. A possible interpretation is that music characterized by few local changes in pitch or rhythm often involves few, highly contrasting pitch-based or rhythmic structural boundaries. For instance, rhythmically stable melodies are clearly separated by highly discernible rests and long notes in *Dvořák*. This could be interpreted in the light of theoretical approaches to musical expectation (Narmour, 1992), according to which similarity between successive events generates the expectation of another similar event. If eventually this expectation is not satisfied, a sense of closure may be perceived, prompting the indication of a segment boundary; this may explain why, for example, accurate tonal-based segmentation predictions exhibit low variability at a local level and often involve few, perceptually stark boundaries that delimit homogeneous groupings of events.

Related to our previous result, we found that novelty curve characteristics can be used as predictors of accuracy: larger distance between subsequent novelty peaks was found to result in higher novelty accuracy (Table 1), especially for tonal features. As aforementioned, this relationship relates to the properties of “good” organizations proposed by Gestalt theorists (Koffka, 1935). In this sense, music characterized by novelty peaks that are clearly isolated should yield higher accuracy as they would relate to perceptually salient musical changes.

We should highlight that the features yielding highest correlations with accuracy were tonal. It is possible that interpretations derived via perceptual organization rules and expectation violation are better applicable to the case of prediction via tonal features because these features focus unambiguously on changes in perceived tonal context (and not on e.g. loudness changes). In contrast, other descriptions used are somewhat more vague: i) Subband Flux discontinuities encompass changes of instrumentation, register, voicing, articulation, and loudness; ii) Fluctuation Pattern changes could be attributed to rhythmic patterns, tempo, articulation, and use of repetition; iii) changes in Chromagram are manifested in pitch steps, pitch jumps, and use of chords. In this respect, tonal features consider a single dimension of musical change, whereas other features analyzed in this study may yield more intricate descriptions.

Accuracy seemed to increase for stimuli with little local change and more distance between peaks (Table 1), however Subband Flux seemed to yield an opposite trend. In this regard, high local variability of changes in instrumentation, register, loudness, etc., seems to be associated with higher accuracy. It could be the case that musical pieces with high local spectral change and multiple novelty peaks are also characterized by few structural sections of long duration, and yield a relatively straightforward prediction; for instance, *Genesis* contains multiple short sounds and effects, yet its sections are clearly delimited by important instrumentation changes, which probably had a positive effect on accuracy. Again, stylistic information might help to disentangle these and other problems regarding segmentation accuracy.

4.3 General Discussion

One of the main aims of this study was to find out methods to select musical features that would be efficient in segmentation prediction for a given stimulus; to this end, we investigated the relationship between accuracy and characterizations of musical features and novelty curves. According to the results, for most features there is an inverse relationship between local variability and accuracy, and a direct relationship between mean distance between subsequent novelty peaks and accuracy. This suggests that stimuli whose features are characterized by low variation between successive time points, and whose novelty curves have few peaks, are likely to yield higher segmentation prediction accuracy. A possible reason that explains these results is that music with infrequent musical change often yields perceptually salient boundaries; according to the Gestalt rules of perceptual organization, similar events that are proximal in time are grouped together, creating a strong sense of closure whenever a dissimilar event is perceived. Following this interpretation, if a given musical dimension changes frequently, an increase of listeners’ attention towards other dimensions evoking strong closure may occur during segmentation.

4.4 Considerations for Further Research

Since this study focused on the analysis of segmentations of the same musical pieces from multiple listeners, the number of segmented stimuli does not suffice to draw solid conclusions about the correlates of segmentation accuracy; this should be considered a major methodological caveat. More musical stimuli are clearly needed to assess the generalization ability of our results; future studies should in this respect increase the number of musical stimuli used for perceptual segmentation tasks while maintaining a satisfactory participant sample size. As regards the sample of participants used, this study only focused on musician listeners, mainly following MIR studies, which recruit expert annotators for the preparation of musical structure data sets. Further work should concentrate on annotation segmentation from nonmusicians in order to understand accuracy of novelty curves with regards to the majority of the population.

Another issue to consider is that the novelty detection approach is designed to yield maximum scores for high continuity within segments and high discontinuity at segmentation points, so in this sense it is not surprising that music exhibiting clear discontinuity between large sequences of homogeneity for a given feature will yield higher segmentation accuracy for that feature. In this regard, our results should be further tested using other approaches; for instance probabilistic methods (e.g. Pauwels, Kaiser, &

Peeters, 2013; Pearce & Wiggins, 2006) would be suitable as they offer alternative assumptions regarding location of actual boundaries.

Finally, as an outcome of this study it can be stated that listeners may tend to focus on musical dimensions that do not change often. This interpretation is plausible and highlights the importance of e.g. tonal and tempo stability, as well as the role of repetition and motivic similarity in musical pieces. Future work should test this possibility by conducting listening studies in which listeners would describe what is the most salient dimension for different time points in the music; further, as suggested by Müller et al. (2016), automatic detection of these acoustic description cues should also be a relevant task regarding structural segmentation.

5. REFERENCES

Bruderer, M. (2008). *Perception and modeling of segment boundaries in popular music* (Doctoral dissertation, JF Schouten School for User-System Interaction Research, Technische Universiteit Eindhoven, Eindhoven).

Cannam, C., Landone, C., & Sandler, M. (2010). Sonic Visualiser: an open source application for viewing, analysing, and annotating music audio files. In *Proceedings of the ACM Multimedia International Conference* (pp. 1467–1468). Firenze, Italy.

Cohen, J. (1988). *Statistical power analysis for the behavioral sciences.* (2nd). New Jersey: Lawrence Erlbaum.

Eronen, A. (2007). Chorus detection with combined use of MFCC and chroma features and image processing filters. In *Proceedings of the 10th International Conference on Digital Audio Effects* (pp. 229–236). Citeseer. Bordeaux.

Foote, J. T. (2000). Automatic audio segmentation using a measure of audio novelty. In *IEEE International Conference on Multimedia and Expo* (Vol. 1, pp. 452–455). IEEE. New York.

Gaudefroy, C., Papadopoulos, H., & Kowalski, M. (2015). A multi-dimensional meter-adaptive method for automatic segmentation of music. In *13th International Workshop on Content-Based Multimedia Indexing (CBMI)* (pp. 1–6). IEEE.

Hartmann, M., Lartillot, O., & Toiviainen, P. (2016a). Interaction features for prediction of perceptual segmentation: effects of musicianship and experimental task. *Journal of New Music Research*.

Hartmann, M., Lartillot, O., & Toiviainen, P. (2016b). Multi-scale modelling of segmentation: effect of musical training and experimental task. *Music Perception*, *34*(2), 192–217.

Jensen, K. (2007). Multiple scale music segmentation using rhythm, timbre, and harmony. *EURASIP Journal on Applied Signal Processing*, *2007*(1), 159–159.

Kaiser, F. & Peeters, G. (2013). Multiple hypotheses at multiple scales for audio novelty computation within music. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*. Vancouver.

Koffka, K. (1935). *Principles of Gestalt psychology*. New York: Harcourt, Brace and Company.

Lartillot, O. & Toiviainen, P. (2007). A Matlab toolbox for musical feature extraction from audio. In *International Conference on Digital Audio Effects* (pp. 237–244). Bordeaux.

Lukashevich, H. M. (2008). Towards quantitative measures of evaluating song segmentation. In *Proceedings of the 9th International Conference on Music Information Retrieval* (pp. 375–380).

McFee, B. & Ellis, D. P. (2014). Learning to segment songs with ordinal linear discriminant analysis. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 5197–5201).

Mendoza Garay, J. (2014). *Self-report measurement of segmentation, mimesis and perceived emotions in acousmatic electroacoustic music* (Master's thesis, University of Jyväskylä).

Müller, M., Chew, E., & Bello, J. P. (2016). Computational Music Structure Analysis (Dagstuhl Seminar 16092). *Dagstuhl Reports*, *6*(2), 147–190. doi:http://dx.doi.org/10.4230/DagRep.6.2.147

Narmour, E. (1992). *The analysis and cognition of melodic complexity: the implication-realization model*. University of Chicago Press.

Paulus, J. & Klapuri, A. (2006). Music structure analysis by finding repeated parts. In *Proceedings of the 1st ACM workshop on audio and music computing multimedia* (pp. 59–68).

Pauwels, J., Kaiser, F., & Peeters, G. (2013). Combining harmony-based and novelty-based approaches for structural segmentation. In *Ismir* (pp. 601–606).

Pearce, M. T. & Wiggins, G. (2006). The information dynamics of melodic boundary detection. In *Proceedings of the ninth international conference on music perception and cognition* (pp. 860–865).

Peeters, G. (2004). Deriving musical structures from signal analysis for music audio summary generation: "sequence" and "state" approach. *Computer Music Modeling and Retrieval*, 169–185.

Peiszer, E., Lidy, T., & Rauber, A. (2008). Automatic audio segmentation: segment boundary and structure detection in popular music. In *Proceedings of the 2nd International Workshop on Learning the Semantics of Audio Signals (LSAS)*. Paris, France.

Pyper, B. J. & Peterman, R. M. (1998). Comparison of methods to account for autocorrelation in correlation analyses of fish data. *Canadian Journal of Fisheries and Aquatic Sciences*, *55*(9), 2127–2140.

Serrà, J., Muller, M., Grosche, P., & Arcos, J. (2014). Unsupervised music structure annotation by time series structure features and segment similarity. *IEEE Transactions on Multimedia*, *16*(5), 1229–1240.

Silverman, B. W. (1986). *Density estimation for statistics and data analysis*. Boca Raton: CRC press.

Smith, J. B. & Chew, E. (2013). A meta-analysis of the mirex structure segmentation task. In *Proc. of the 14th International Society for Music Information Retrieval Conference* (Vol. 16, pp. 45–47). Curitiba.

Meter Detection in Symbolic Music Using a Lexicalized PCFG

Andrew McLeod
University of Edinburgh
A.McLeod-5@sms.ed.ac.uk

Mark Steedman
University of Edinburgh
steedman@inf.ed.ac.uk

ABSTRACT

This work proposes a lexicalized probabilistic context free grammar designed for meter detection, an integral component of automatic music transcription. The grammar uses rhythmic cues to align a given musical piece with learned metrical stress patterns. Lexicalization breaks the standard PCFG assumption of independence of production, and thus, our grammar can model the more complex rhythmic dependencies which are present in musical compositions. Using a metric we propose for the task, we show that our grammar outperforms baseline methods when run on symbolic music input which has been hand-aligned to a tatum. We also show that the grammar outperforms an existing method when run with automatically-aligned symbolic music data as input. The code for our grammar is available at https://github.com/apmcleod/met-detection.

1. INTRODUCTION

Meter detection is the organisation of the beats of a given musical performance into a sequence of trees at the bar level, in which each node represents a single note value. In common-practice Western music (the subject of our work), the children of each node in the tree divide its duration into some number of equal-length notes (usually two or three) such that every node at a given depth has an equal value. For example, the metrical structure of a single $\frac{3}{4}$ bar, down to the quaver level, is shown in Fig. 1. Additionally, the metrical structure must be properly aligned in phase with the underlying musical performance so that the root of each tree corresponds to a single bar. Each level of a metrical tree corresponds with an isochronous pulse in the underlying music: bar, beat, and sub-beat (from top to bottom). There are theoretically further divisions further down the tree, but as these three levels are enough to unambiguously identify the meter of a piece, we do not consider any lower.

The task is an integral component of Automatic Music Transcription (AMT), particularly when trying to identify the time signature of a given performance, since there is a one-to-one relationship between time signatures and metrical structures. In music, each successive bar may have a different metrical structure than the preceding one; however, such changes in structure are not currently handled by our model, and are left for future work. Our grammar can only be applied to pieces in which each bar is of equal length, has the same number of equal-length beats, and each beat has the same number of equal-length sub-beats. That is, it can be applied to any piece where the metrical tree structure under each node at a given level of the tree is identical. In this work, we evaluate our grammar only on the simple and compound meter types $\frac{2}{X}$, $\frac{3}{X}$, $\frac{4}{X}$, $\frac{6}{X}$, $\frac{9}{X}$, and $\frac{12}{X}$ (where X can be any of 1, 2, 4, 8, or 16), and leave more uncommon and irregular meters for future work. Those interested in asymmetric meter detection should refer to [1].

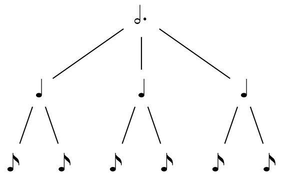


Figure 1. The metrical structure of a $\frac{3}{4}$ bar.

Our grammar is designed to be run on symbolic music data such as MIDI. In this work, we present two experiments: one where the tatum—the fastest subdivision of the beat (we use demisemiquavers, or 32nd notes)—is given, and another where a beat-tracker is used as a preprocessing step to automatically detect some tatum (not necessarily demisemiquavers). Note that ideally, the beat-tracker would be run jointly with our grammar, as beat and meter are intrinsically related; however, we leave such a joint model for future work.

Thus, the task that our grammar solves is one of identifying the correct full metrical structure, composed of: (1) meter type (the number of beats per bar and the number of sub-beats per beat), (2) phase (the number of tatums which fall before the first full bar), and (3) sub-beat length (the number of tatums which lie within a single sub-beat).

2. EXISTING WORK

Most of the early work in the field of meter detection involved rule-based, perceptual models. Longuet-Higgins and Steedman [2] present one which runs on monophonic quantized data and uses only note durations, which was later extended by Steedman [3] to incorporate melodic repetition. Both models were evaluated on full metrical structure detection on the fugues from Bach's Well-Tempered Clavier (WTC). Longuet-Higgins and Lee [4] describe a somewhat similar model, also to be run on monophonic quantized data, though only a few qualitative examples are presented in evaluation, and the model is unable to handle syncopation. Spiro [5] proposes a rule-based, incremental model for quantized data, combined with a probabilistic n-gram model of bar-length rhythmic patterns, and evaluated on full metrical structure detection on a small corpus of 16 monophonic string compositions by Bach. This remains one of the only successful models for meter detection to use a grammar thus far, though similar grammars have been used for rhythmic and tempo analysis where the meter is given [6–8]. While these rule-based methods show promise, and we base some of our model's principals on them, a more flexible probabilistic model is preferred.

Brown [9] proposed using auto-correlation for meter detection, in which a promising, though limited, evaluation on meter type and sub-beat length detection was shown for 17 quantized pieces, Meudic [10] later proposed a similar model also using auto-correlation on quantized MIDI data for the same task. Eck and Casagrande [11] extended this further, and were the first to use auto-correlation to also calculate the phase of the meter (though phase detection results are limited to synthetic rhythms). They were also the first to do some sort of corpus-based evaluation, though only to classify the meter of a piece as duple or compound. Though auto-correlation has performed well for partial metrical structure detection, there is still a question about whether it can detect the phase of that meter, and no work that we have found has yet done so successfully from real symbolic music data.

Inner Metric Analysis (IMA) was first proposed for music analysis by Volk [12], though only as a method to analyse the rhythmic stress of a piece, not to detect the meter of that piece. It requires quantized MIDI with labeled beats as input, and it involves identifying periodic beats which align with note onsets. Thus, detecting metrical structure and phase using IMA is a matter of classifying the correct beats as downbeats; it is used by De Haas and Volk [13], along with some post-processing, to perform meter detection on quantized MIDI data probabilistically. We were unable to run their model on our data, though they evaluate the model on two datasets, testing both duple or triple classification as well as full metrical structure detection (including phase). However, as the datasets they used are quite homogeneous—95% of the songs in the FMPop corpus are in $\frac{4}{4}$, and 92% of the songs in the RAG corpus [14] are in either $\frac{2}{4}$ or $\frac{4}{4}$ time—we have decided not to include a comparison in this work.

Whiteley et al. [15] perform full metrical structure detection probabilistically from live performance data by jointly modeling tempo, meter, and rhythm; however, the evaluation was very brief, only testing the model on 3 bars of a single Beatles piano performance, and the idea was not used further on symbolic data to our knowledge. Temperley [16] proposes a Bayesian model for the meter detection of unquantized, monophonic MIDI performance data. The general idea is to model the probability of a note onset occurring given the current level of the metrical tree at any time with Bayes' rule. This is combined with a simple Bayesian model of tempo changes, giving a model which can detect the full metrical structure of a performance. Temperley [17] extends this model to work on polyphonic data, combining it into a joint model with a Bayesian voice separator and a Bayesian model of harmony. This joint model performs well on full metrical structure detection on a corpus of piano excerpts, and we compare against it in this work.

$$
\begin{array}{ll}
S & \rightarrow M_{b,s} \\
M_{b,s} & \rightarrow B_s \ldots B_s \ (b \text{ times}) \\
B_s & \rightarrow SB \ldots SB \ (s \text{ times}) \mid r \\
SB & \rightarrow r
\end{array}
$$

Figure 2. The grammar rules which form the basis of the PCFG. The subscript b is the number of beats per bar, while s is the number of sub-beats per beat. The terminal symbol r can refer to any rhythmic pattern.

3. PROPOSED METHOD

For our proposed method, we were careful to make as few assumptions as possible so it can be applied to different styles of music directly (assuming enough training data is available). It is based on a standard probabilistic context free grammar (PCFG; presented in Section 3.1) with added lexicalization as as introduced in Section 3.2. The inference procedure is described in Section 3.3.

The basic idea of the grammar is to detect patterns of rhythmic stress in a given piece of music with the grammar, and then to measure how well those stress patterns align with metrical stress patterns. We use note length to measure rhythmic stress in this work, assuming that long notes will be heard as stressed. This assumption is based on ideas from many of the rule-based methods presented above, and works well; however, there are many other factors of musical stress that our grammar does not capture, such as melody and harmony, which have been found to be helpful for meter detection [18], and will be incorporated into our grammar in future work.

3.1 PCFG

The context-free grammar shown in Fig. 2 is used to construct a rhythmic tree quite similar to the metrical tree from Figure 1 above. Each bar of a given piece is first assigned the start symbol S, which can be rewritten as the non-terminal $M_{b,s}$ (representing the meter type), where b is the number of beats per bar and s is the number of sub-beats per beat (2 for simple meters and 3 for compound meters). For example, $M_{4,2}$ represents a meter in $\frac{4}{4}$ time, and $M_{2,3}$ represents a meter in $\frac{6}{8}$ time.

A non-terminal $M_{b,s}$ is rewritten as b beat non-terminals B_s. Each beat non-terminal B_s can be rewritten either as s sub-beat non-terminals SB or as the terminal r, representing the underlying rhythm of the beat. A beat may only be rewritten as r if it contains either (1) no notes or (2) a single note which lasts at least the entire duration of the node (the note may begin before the beat, end after the beat, or both). A sub-beat SB must be rewritten as a terminal r, representing the underlying rhythm of that sub-beat.

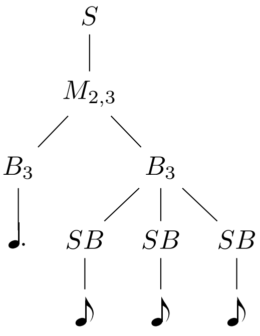


Figure 3. An example of the rhythmic tree of a $\frac{6}{8}$ bar with the rhythm ♩. ♪♪♪.

An example of the rhythmic tree of a single $\frac{6}{8}$ bar with the rhythm ♩. ♪♪♪ is shown in Figure 3. Here, the first beat is been rewritten as a terminal since it is the only note present.

3.2 Lexicalization

One downside of using a PCFG to model the rhythmic structure is that PCFGs make a strong independence assumption that is not appropriate for music. Specifically, in a given rhythm, a note can only be heard as stressed or important in contrast with the notes around it, though a standard PCFG cannot model this. A PCFG may see a dotted quarter note and assume that it is a long note, even though it has no way of knowing whether the surrounding notes are shorter or longer, and thus, whether the note should indeed be considered stressed.

To solve this problem, we implement a lexicalized PCFG (LPCFG), where each terminal is assigned a head corresponding to its note with the longest duration. Strong heads (in this work, those representing longer notes) propagate upwards through the metrical tree to the non-terminals in a process called lexicalization. This allows the grammar to model rhythmic dependencies rather than assuming independence as in a standard PCFG, and the pattern of strong and weak beats and sub-beats is used to determine the underlying rhythmic stress pattern of a given piece of music.

This head is written $(d; s)$, where d is the duration of the longest note (or, the portion of that note which lies beneath the node), and s is the starting position of that note. The character 't' is added to the end of s if that note is tied into (i.e. if the onset of the note lies under some previous node). In the heads, d and s are normalized so that the duration of the node itself is 1. Thus, only heads which are assigned to nodes at the same depth can be compared directly. A node with no notes is assigned the empty head of $(0; 0)$.

Once node heads have been assigned, each beat and sub-beat non-terminal is assigned a strength of either strong (S), weak (W), or even (E). These are assigned by comparing the heads of siblings in the rhythmic tree. If all siblings' heads are equal, they are assigned even strength. Otherwise, those siblings with the strongest head are assigned strong strength while all others are assigned weak strength, regardless of their relative head strengths.

Head strength is determined by a ranking system, where heads are first ranked by d such that longer notes are considered stronger. Any ties are broken by s such that an earlier starting position corresponds to greater strength. Any further ties are broken such that notes which are not tied into are considered stronger than those which are.

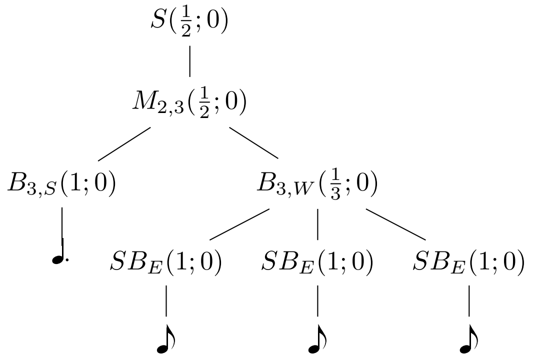


Figure 4. An example of the rhythmic tree of a $\frac{6}{8}$ bar with the rhythm ♩. ♪♪♪ including strengths and lexicalization.

An example of the rhythmic tree of a single $\frac{6}{8}$ bar with the rhythm ♩. ♪♪♪ including strengths and lexicalization is shown in Figure 4.

3.3 Performing Inference

Each of the LPCFG rule probabilities are computed as suggested by Jurafsky and Martin [19], plus additionally conditioning each on the meter type. For example, the replacement $\{M_{2,3}(\frac{1}{2}; 0) \rightarrow B_{3,S}(1; 0)\ B_{3,W}(\frac{1}{3}; 0)\}$ is modeled by the product of Equations (1), (2), and (3). Equation (1) models the probability of a transition given the left-hand-side node's head, while Equations (2) and (3) model the probability of each child's head given its type and the parent's head.

$$p(M_{2,3} \rightarrow B_{3,S}\ B_{3,W} \mid M_{2,3}, (1/2; 0)) \quad (1)$$

$$p((1; 0) \mid M_{2,3}, B_{3,S}, (1/2; 0)) \quad (2)$$

$$p((1/3; 0) \mid M_{2,3}, B_{3,W}, (1/2; 0)) \quad (3)$$

The meter type conditioning ensures that the model not prefer one meter type over another based on uneven training data. Specifically, each initial transition $S \rightarrow M_{b,s}$ is assigned a probability of 1. The actual probability values are computed given a training corpus using maximum likelihood estimation with Good-Turing smoothing as described by Good [20]. If a given replacement's head, as modeled by Equations (2) and (3), is not seen in the training data, we use a backing-off technique as follows. We multiply the probability from the Good-Turing smoothing by a new probability equation, where the meter type is removed (again with Good-Turing smoothing). This allows the grammar to model, for example, the beat-level transitions of a $\frac{9}{8}$ bar using the beat-level transitions of a $\frac{3}{4}$ bar. Note that this does not allow any cross-level calculations where, for example, the beat level of a $\frac{9}{8}$ bar could be modeled by the sub-beat level of a $\frac{6}{8}$ bar, though this could be a possible avenue for future work.

The grammar was designed to be used on monophonic melodies, so we use the voices as annotated in the data. Afterwards, only rhythmic information is needed. That is, the grammar uses onset and offset times for each note, and no pitch or velocity information.

The first step in the inference process is to create multiple hypothesis states, each with probability 1, and each corresponding to a different (meter type, sub-beat length, phase) triplet, which are treated as latent variables. Meter type corresponds to the specific $M_{b,s}$ which will be used throughout the piece for that hypothesis (there is currently a constraint that pieces do not change time signature during a piece). Sub-beat length corresponds to the length of a sub-beat of that hypothesis state. This differentiates $\frac{2}{4}$ time from $\frac{2}{2}$ time, for example. Phase refers to how long of an anacrusis a hypothesis will model. That is, how many tatums lie before the first full bar.

Each state's rhythmic trees are built deterministically, one per voice per bar while that voice is active, throwing out any anacrusis bars. A state's final probability is the product of the probabilities of each of the trees of each of its voices. After running through a full piece, the states are ordered by probability and the metrical structure corresponding to the most likely state's (meter type, sub-beat length, phase) triplet is picked as the model's guess.

One final optimization is made, related to the "rule of congruence" as noted by Longuet-Higgins and Steedman [2], and further described perceptually by Lee [21]. That is, with few exceptions, a composer (at least of classical music), will not syncopate the rhythm before a meter has been established. This means that if the meter has not yet been established, and the underlying rhythm does not match with the metrical structure of a hypothesis state based on its (meter type, sub-beat length, phase) triplet, we should be able to remove it. In practice, we allow up to 5 mismatches before eliminating a metrical hypothesis state. In tests, setting this value to anything from 2 to 20 makes no difference, just the lower the value the faster the program becomes (and the less room for error there is in the case of a particularly adventurous composer). For full details on the implementation of this rule, see Appendix A.

4. EVALUATION

4.1 Metric

To evaluate our method, instead of just checking whether the top hypothesis' metrical structure is fully correct or not, we wanted some measure of partial correctness. For instance, if the correct time signature is $\frac{4}{4}$, a guess of $\frac{2}{4}$ should achieve a higher score than a guess of $\frac{6}{8}$. With that in mind, we propose the following metric.

For each of the three levels of the guessed metrical structure, an exact match with a level of the correct metrical structure is counted as a true positive, while a clash—when all of the nodes in a level of the guessed structure cannot be made by some integer multiple or division of nodes from each of the levels of the correct structure—is counted as a false positive. After all three levels have been tested, each of the correct metrical structure's levels which were not matched count as a false negative. Precision, recall, and F1 can all be computed based on the resulting true positive, false positive, and false negative totals.

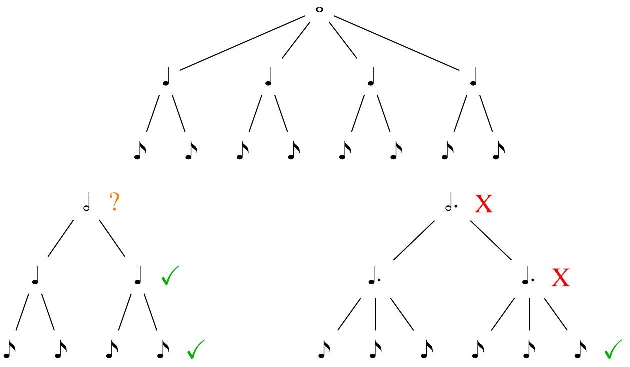


Figure 5. Top: The metrical structure of a $\frac{4}{4}$ bar. If $\frac{4}{4}$ is the correct time signature, a guess of $\frac{2}{4}$ with the correct phase (bottom-left) would give $P = 1.0$, $R = 0.67$, and $F1 = 0.8$. A guess of $\frac{6}{8}$ with the correct phase (bottom-right) would give $P = 0.33$, $R = 0.33$, and $F1 = 0.33$.

Examples of this metric are illustrated in Fig. 5. Given a correct time signature of $\frac{4}{4}$, and assuming that the phase of the guessed metrical structure is correct, if the guessed time signature is $\frac{2}{4}$, there are only 2 true positives; however, the bar-level grouping for $\frac{2}{4}$ does not clash with the metrical structure of $\frac{4}{4}$, so it is not counted as a false positive. There is, however, 1 false negative from the bar level of the $\frac{4}{4}$, giving values of $P = 1.0$, $R = 0.67$, and $F1 = 0.8$. If $\frac{6}{8}$ is guessed instead, the sub-beat level again matches, giving 1 true positive. However, both the beat level and the bar level clash (since 1.5 beats of a $\frac{4}{4}$ bar make a single $\frac{6}{8}$ beat, and $\frac{3}{4}$ of a $\frac{4}{4}$ bar gives a $\frac{6}{8}$ bar), giving 2 false positives and 2 false negatives. This gives values of $P = 0.33$, $R = 0.33$, and $F1 = 0.33$. Much lower, and rightfully so.

For evaluation on a full corpus, true positives, false positives, and false negatives are summed throughout the entire corpus to get a global precision, recall, and F1.

4.2 Data

We report our results on two main corpora: (1) the 15 Bach Inventions, consisting of 1126 monophonic bars (in which a single bar with two voices counts as two bars), and (2) the much larger set of 48 fugues from the Well-Tempered Clavier, containing 8835 monophonic bars. These two corpora contain quantized MIDI files, hand-aligned with a demisemiquaver (32nd note) tatum, and we present results using both this hand-aligned tatum and an automatically-aligned tatum. The notes in each file are split into voices as marked in the corresponding scores. We present additional evaluation in the automatically-aligned case using the German subset of the Essen Folksong Collection [22], 4954 pieces in total, consisting of 66356 monophonic bars.

We use leave-one-out cross-validation within each corpus for learning the probabilities of the grammar. That is, for testing each song in a corpus, we train our grammar on all of the other songs within that corpus. We also tried using

Method	Inventions	Fugues
$\frac{4}{4}$	0.58	0.45
PCFG	0.61	0.63
LPCFG	**0.63**	**0.80**

Table 1. The F1 of each method for each corpus using a hand-aligned tatum.

Method	Inventions	Fugues	Essen [22]
Temperley [17]	**0.58**	0.63	0.60
LPCFG+BT	0.55	**0.72**	**0.74**

Table 2. The F1 of each method for each corpus using an automatically-aligned tatum.

Meter	Inventions			Fugues		
Type	#	LPCFG	+BT	#	LPCFG	+BT
$\frac{6}{X}$	0	—	—	4	0.58	0.83
$\frac{3}{X}$	5	0.60	0.64	7	0.57	0.88
$\frac{2}{X}$	0	—	—	9	0.89	0.76
$\frac{4}{X}$	8	0.71	0.55	26	**0.90**	0.66
All	15	0.63	0.55	48	0.80	0.72

Table 3. The F1 of the LPCFG and LPCFG+BT split by meter type. Here, # represents the number of pieces in each meter type, and meter types which occur only once in a corpus are omitted.

cross-validation across corpora by training on the Inventions when testing the Fugues and vice versa; however, that led to similar but slightly worse results, as the complexities of the rhythms in the corpora are not quite similar enough to allow for such training to be successful. Specifically, there is much more syncopation in the Fugues than in the Inventions, and thus our grammar would tend to prefer to incorrectly choose meters for the Inventions which would result in some syncopation.

4.3 Results

With hand-aligned input, the LPCFG is evaluated against two baselines. First, a naive one, guessing $\frac{4}{4}$ time with an anacrusis such that the first full bar begins at the onset of the first note (the most common time signature in each corpus). Second, the PCFG without lexicalization (as proposed in Section 3.1, with Good-Turing smoothing and rule of congruence matching).

With automatically-aligned input, we evaluate against the model proposed by Temperley [17], which performs beat tracking jointly with meter detection. For direct comparison, we use the fastest beat given by Temperley's model as the tatum, and we call this version of our grammar an LPCFG with beat tracking (LPCFG+BT). It would be better to perform beat tracking and meter detection jointly, as in Temperley's model; however, we leave such joint inference for future work. The automatic alignment presents some difficulty in computing our metric, since a metrical hypothesis generated from an incorrectly aligned input may move in and out of phase throughout the course of a piece due to beat tracking errors. Therefore, we evaluate each meter based on its phase and sub-beat length relative only to the first note of each piece, thus avoiding any misalignments caused by subsequent errors in beat tracking.

The results for hand-aligned input can be found in Table 1, where it can be seen that the LPCFG outperforms all baselines, quite substantially on the fugues. The results for automatically-aligned input are shown in Table 2. Here, the LPCFG+BT outperforms Temperley's model on the fugues and the Essen corpus, but is outperformed by it on the inventions.

For both hand-aligned and automatically-aligned input, it is surprising that the LPCFG (and LPCFG+BT) does not perform better on the inventions, which are simpler compositions than the fugues. The reason for this lack of improvement seems to be a simple lack of training data, as can be seen in Table 3, which shows that as the number of training pieces for each meter type increases, the performance of the LPCFG improves dramatically, though the LPCFG+BT does not follow this trend.

During automatic tatum alignment, beat-tracking tends to quantize rare patterns (which may occur only a single meter type) into more common ones, allowing the grammar to identify the rhythmic stress of a piece without having to parse too many previously unseen rhythms. This helps in the case of not enough training data, but can hurt when more training data is available. These rare patterns tend to be strong markers of a certain meter type, and if enough training data is available to recognize them, such quantization would remove a very salient rhythmic clue. Therefore, for both hand-aligned and automatically-aligned input, more training data in the style of the inventions should continue to improve its performance on that corpus.

Fig. 6 shows the percentage of pieces in each corpus for which each method achieves 3, 2, 1, or 0 true positives, and further details exactly what is happening on the inventions. The true positive counts correspond with those in our metric, and represent the number of levels of the metrical tree (bar, sub-beat, beat) which were matched in both length and phase for each piece. Thus, more true positives corresponds with a more accurate guess.

The improvement on the fugues is clear for both types of input. On the hand-aligned inventions, however, the naive $\frac{4}{4}$ model gets 40% of the metrical structures of the inventions exactly correct (3 TPs), while the LPCFG gets only 26.67%. The LPCFG gets significantly more of its guesses mostly or exactly correct (with 2 or 3 TPs), and eliminates totally incorrect guesses (0 TPs) completely. This shows that, even though it may not have had enough data yet to classify time signatures correctly, it does seem to be learning some sort of structural patterns from what limited data it has. Meanwhile, on the automatically-aligned inventions, the LPCFG+BT gets slightly fewer of its guesses mostly correct than Temperley's model, but significantly fewer (only one invention) totally incorrect, again showing that it has learned some structural patterns. The slightly lower F1 of the LPCFG+BT on the automatically-aligned inventions is due to a higher false positive rate than Temperley's model.

That our model's performance is more sensitive to a lack

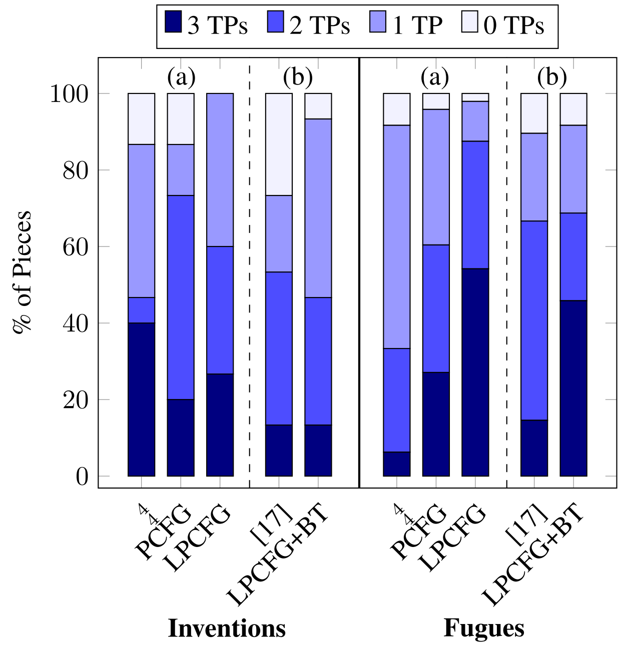


Figure 6. The percentage of pieces in each corpus for which each method's metrical structure guess resulted in the given number of true positives, for (a) hand-aligned and (b) automatically-aligned beats.

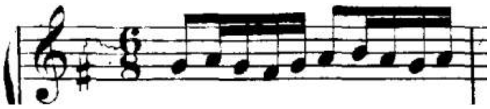


Figure 7. The first bar of 15th fugue from WTC book I by Bach (BWV 860).

of training data is further illustrated by the fact that our model outperforms Temperley's substantially on the Essen corpus, where ample training data is available. Indeed, in general, the LPCFG+BT performs worse than Temperley's model when there is a lack of training data, but outperforms it when enough data exists.[1]

A specific case where increased training data would benefit the LPCFG is in the 15th fugue from WTC book I, the first bar of which is shown in Fig. 7. This rhythmic pattern is found throughout the piece, and is a strong indication of a $\frac{6}{8}$ bar, consisting of two even beats, each split into a sub-beat pattern of strong, weak, weak. However, our grammar guesses that this piece is in $\frac{4}{4}$ time simply because it has not seen the transition $\{B_{3,E} \rightarrow SB_S\, SB_W\, SB_W\}$ in a $\frac{6}{X}$ meter type in training. This is indicative of the errors we see throughout the data, showing again that with more training data the results will only improve.

5. CONCLUSION

In this paper, we have proposed an LPCFG for full metrical structure detection of symbolic music data. We have shown that this grammar improves over multiple baseline methods when run on hand-aligned symbolic music input, and that it can be combined with a beat tracking model to achieve good meter detection results on automatically-aligned symbolic music data. The fact that lexicalization adds definite value over a standard PCFG shows that there are complex rhythmic dependencies in music which such lexicalization is able to capture.

Our model is somewhat sensitive to a lack of training data, though it does learn metrical stress patterns quickly, and we will also look at more aggressive cross-level back-off techniques to make the grammar more robust to such a lack of data. For example, it may be possible to model the transitions at the sub-beat level of a $\frac{9}{X}$ meter type using the beat level transitions of a $\frac{3}{X}$ meter type. Furthermore, we will also look to apply our model to more uncommon or irregular meter types such as $\frac{5}{X}$ or $\frac{7}{X}$, perhaps as the concatenation of the more common meter types.

The proposed LPCFG shows promise in meter detection even using only rhythmic data, and future work will incorporate melodic and harmonic information into the grammar. For example, harmonic changes are most likely to occur at the beginnings of bars, and low notes occur more frequently on strong beats, suggesting that incorporating pitch and harmony into the lexical heads may improve performance. Without such additions, our grammar is helpless to hypothesize a meter for an isochronous melody.

Another avenue of future work is to adapt the grammar for use on live performance data by performing inference on the grammar jointly with a beat-tracker. This is more natural than performing beat-tracking as a preprocessing step, as beat and meter are closely related. We will also consider the grammar's application to acoustic data; we have run preliminary experiments using off-the-shelf onset detection models, but found that a more complex approach is needed. Specifically, some sort of voicing information for the onsets would improve performance dramatically, since it would give a more accurate measurement of the lengths of the notes corresponding to each onset.

6. ACKNOWLEDGEMENTS

This work was partly funded through a gift from the 2017 Bloomberg Data Science Research Grant program.

7. REFERENCES

[1] T. Fouloulis, A. Pikrakis, and E. Cambouropoulos, "Traditional Asymmetric Rhythms: A Refined Model of Meter Induction Based on Asymmetric Meter Templates," in *Proceedings of the Third International Workshop on Folk Music Analysis*, 2013, pp. 28–32.

[2] H. C. Longuet-Higgins and M. Steedman, "On Interpreting Bach," *Machine Intelligence*, vol. 6, pp. 221–241, 1971.

[3] M. Steedman, "The Perception of Musical Rhythm and Metre," *Perception*, vol. 6, pp. 555–569, jan 1977.

[4] H. C. Longuet-Higgins and C. S. Lee, "The perception of musical rhythms," *Perception*, vol. 11, no. 2, pp. 115–128, 1982.

[1] We do not include evaluation on the Essen corpus with hand-aligned tatum because the pieces are very short and quite simple rhythmically.

[5] N. Spiro, “Combining Grammar-Based and Memory-Based Models of Perception of Time Signature and Phase,” in *Music and Artificial Intelligence*. Springer Berlin Heidelberg, 2002, pp. 183–194.

[6] H. Takeda, T. Nishimoto, and S. Sagayama, “Rhythm and Tempo Recognition of Music Performance from a Probabilistic Approach.” in *ISMIR*, 2004.

[7] ——, “Rhythm and Tempo Analysis Toward Automatic Music Transcription,” in *ICASSP*. IEEE, 2007, pp. 1317–1320.

[8] E. Nakamura, K. Yoshii, and S. Sagayama, “Rhythm Transcription of Polyphonic MIDI Performances Based on a Merged-Output HMM for Multiple Voices,” in *SMC*, 2016.

[9] J. C. Brown, “Determination of the meter of musical scores by autocorrelation,” *The Journal of the Acoustical Society of America*, vol. 94, no. 4, p. 1953, 1993.

[10] B. Meudic, “Automatic Meter Extraction from MIDI Files,” in *Journées d'informatique musicale*, 2002.

[11] D. Eck and N. Casagrande, “Finding Meter in Music Using An Autocorrelation Phase Matrix and Shannon Entropy,” in *ISMIR*, 2005, pp. 504–509.

[12] A. Volk, “The Study of Syncopation Using Inner Metric Analysis: Linking Theoretical and Experimental Analysis of Metre in Music,” *Journal of New Music Research*, vol. 37, no. 4, pp. 259–273, dec 2008.

[13] W. B. de Haas and A. Volk, “Meter Detection in Symbolic Music Using Inner Metric Analysis,” in *ISMIR*, 2016, pp. 441–447.

[14] A. Volk and W. B. de Haas, “A corpus-based study on ragtime syncopation,” *ISMIR*, 2013.

[15] N. Whiteley, A. T. Cemgil, and S. Godsill, “Bayesian Modelling of Temporal Structure in Musical Audio,” in *ISMIR*, 2006.

[16] D. Temperley, *Music and Probability*. The MIT Press, 2007.

[17] ——, “A Unified Probabilistic Model for Polyphonic Music Analysis,” *Journal of New Music Research*, vol. 38, no. 1, pp. 3–18, mar 2009.

[18] P. Toiviainen and T. Eerola, “Autocorrelation in meter induction: The role of accent structure,” *The Journal of the Acoustical Society of America*, vol. 119, no. 2, p. 1164, 2006.

[19] D. Jurafsky and J. H. Martin, “Speech and language processing,” *International Edition*, 2000.

[20] I. J. Good, “The population frequencies of species and the estimation of population parameters,” *Biometrika*, pp. 237–264, 1953.

[21] C. Lee, “The perception of metrical structure: Experimental evidence and a new model,” in *Representing Musical Structure*, P. Howell, R. West, and I. Cross, Eds. Academic Press, May 1991, ch. 3, pp. 59–128.

[22] H. Schaffrath and D. Huron, “The essen folksong collection in the humdrum kern format,” *Menlo Park, CA: Center for Computer Assisted Research in the Humanities*, 1995.

A. RULE OF CONGRUENCE

A metrical structure hypothesis begins as unmatched, and is considered to be fully matched if and only if both its beat and sub-beat levels have been matched. Thus, a metrical hypothesis can be in one of four states: fully matched, sub-beat matched, beat matched, or unmatched.

If a hypothesis is unmatched, a note which is shorter than a sub-beat and does not divide a sub-beat evenly is counted as a mismatch. A note which is exactly a sub-beat in length is either counted as a mismatch (if it is not in phase with the sub-beat), or the hypothesis is moved into the sub-beat matched state. A note which is between a sub-beat and a beat in length is counted as a mismatch. A note which is exactly a beat in length is either counted as a mismatch (if it is not in phase with the beat), or the hypothesis is moved into the beat matched state. A note which is longer than a beat, is not some whole multiple of a beat in length, and does not divide a bar evenly is counted as a mismatch.

If a hypothesis is sub-beat matched, it now interprets each incoming note based on that sub-beat length. That is, any note which is longer than a single sub-beat is divided into up to three separate notes (for meter matching purposes only): (1) The part of the note which lies before the first sub-beat boundary which it overlaps (if the note begins exactly on a sub-beat, no division occurs); (2) The part of the note which lies after the final sub-beat boundary which it overlaps (if the note ends exactly on a sub-beat, no division occurs); and (3) the rest of the note. After this processing, a note which is longer than a sub-beat and shorter than a beat is counted as a mismatch. (Due to note division, this only occurs if the note is two sub-beats in length and each beat has three sub-beats.) A note which is exactly a beat in length moves the hypothesis into the fully matched state. A note which is longer than a beat and is not some whole multiple of beats is counted as a mismatch.

If a hypothesis is beat matched, it now interprets each incoming note based on that beat length exactly as is described for sub-beat length in the previous paragraph. After this processing, a note which is shorter than a sub-beat and does not divide a sub-beat evenly is counted as a mismatch. A note which is exactly a sub-beat in length is either counted as a mismatch (if it is not in phase with the sub-beat), or the hypothesis is moved into the fully matched state. A note which is longer than a sub-beat and shorter than a beat, and which does not align with the beginning or end of a beat, is counted as a mismatch.

Once a metrical hypothesis is fully matched, incoming notes are no longer checked for matching, and the hypothesis will never be removed.

AUDIO-TO-SCORE ALIGNMENT OF PIANO MUSIC USING RNN-BASED AUTOMATIC MUSIC TRANSCRIPTION

Taegyun Kwon, Dasaem Jeong, and Juhan Nam
Graduate School of Culture Technology, KAIST
{ilcobo2, jdasam, juhannam}@kaist.ac.kr

ABSTRACT

We propose a framework for audio-to-score alignment on piano performance that employs automatic music transcription (AMT) using neural networks. Even though the AMT result may contain some errors, the note prediction output can be regarded as a learned feature representation that is directly comparable to MIDI note or chroma representation. To this end, we employ two recurrent neural networks that work as the AMT-based feature extractors to the alignment algorithm. One predicts the presence of 88 notes or 12 chroma in frame-level and the other detects note onsets in 12 chroma. We combine the two types of learned features for the audio-to-score alignment. For comparability, we apply dynamic time warping as an alignment algorithm without any additional post-processing. We evaluate the proposed framework on the MAPS dataset and compare it to previous work. The result shows that the alignment framework with the learned features significantly improves the accuracy, achieving less than 10 ms in mean onset error.

1. INTRODUCTION

Audio-to-score alignment (also known as score following) is the process of temporally fitting music performance audio to its score. The task has been explored for quite a while and utilized mainly for interactive music applications, for example, automatic page turning, computer-aided accompaniment or interactive interface for active music listening [1, 2]. Another use case of audio-to-score alignment is performance analysis which examines performer's interpretation of music pieces in terms of tempo, dynamics, rhythm and other musical expressions [3]. To this end, the alignment result must be sufficiently precise having high temporal resolution. It was reported that the just-noticeable difference (JND) time displacement of a tone presented in a metrical sequence is about 10 ms for short notes [4], which is beyond the current accuracy of the automatic alignment algorithm. This challenge has provided the motivation for our research.

There are two main components in audio-to-score alignment: features used in comparing audio to score, and alignment algorithm between two feature sequences. In this paper, we limit our scope to the feature part. A typical approach is converting MIDI score to synthesized audio and comparing it to performance audio using various audio features. The most common choices are time-frequency representations through short time Fourier transformation (STFT) [5] or auditory filter bank responses [6]. Others suggested chroma audio features, which are designed to minimize differences in acoustic quality between two piano audio such as timbre, dynamics and sustain effects [6]. However, the design process by hands relies on trial-and-error and so is time-consuming and sub-optimal. Another approach to audio-to-score alignment is converting the performance audio to MIDI using automatic music transcription (AMT) systems and comparing the performance to score in the MIDI domain [7]. The advantage of this approach is that the transcribed MIDI is robust to timbre and dynamics variations by the nature of the AMT system if it predicts only the presence of notes. In addition, the synthesis step is not required. However, the AMT system must have high performance to predict notes accurately, which is actually a challenging task.

In this paper, we follow the AMT-based approach for audio-to-score alignment. To this end, we build two AMT systems by adapting a state-of-art method using recurrent neural networks [8] with a few modifications. One system takes spectrograms as input and is trained in a supervised manner to predict a binary representation of MIDI in either 88 notes or chroma. The prediction does not consider intensities of notes, i.e. MIDI velocity. Using this system only however does not provide precise alignment because onset frames and sustain frames are equally important. In order to make up for the limitation, we use another AMT system that is trained to predict the onsets of MIDI notes in chroma domain. This was inspired from Decaying Locally-adaptive Normalized Chroma Onset (DLNCO) feature by Ewert et al. [6]. Following the idea, we employ decaying chroma note onset features which turned out to offer not only temporally precise points but also make onset frames salient. Finally, we combine the two MIDI domain features and conduct dynamic time warping algorithm on the feature similarity matrix. The evaluation on the MAPS dataset shows that our proposed framework significantly improves the alignment accuracy compared to previous work.

2. SYSTEM DESCRIPTION

The proposed framework is illustrated in Figure 1. The left-hand presents the two independent AMT systems that

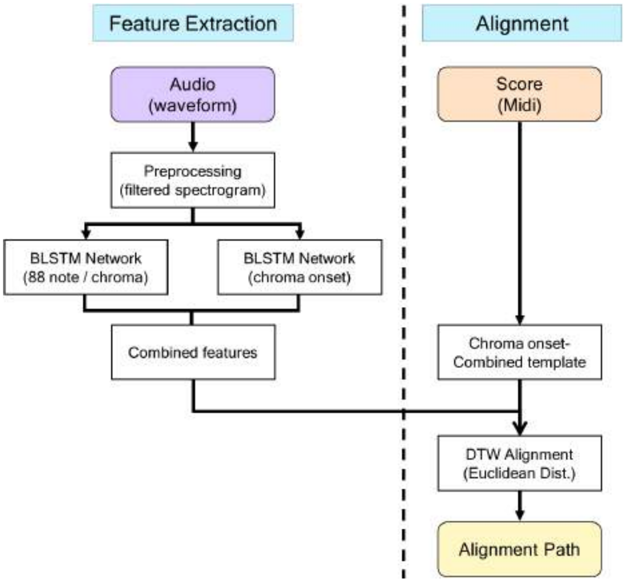

Feature Extraction
Alignment
Audio (waveform)
Preprocessing (filtered spectrogram)
BLSTM Network (88 note / chroma)
BLSTM Network (chroma onset)
Combined features
Score (Midi)
Chroma onset-Combined template
DTW Alignment (Euclidean Dist.)
Alignment Path

Figure 1. Flow diagram of proposed audio-to-score alignment system

return either 88 note or chroma output and chroma onset output, respectively. The outputs are concatenated and aligned with the score MIDI through dynamic time warping (DTW). Since our main idea is not improving the performance of AMT system but rather utilizing a neural-network based system that produces features for audio-to-score alignment, we borrowed the state-of-art AMT system proposed by Böck and Schedl [8]. However, we slighly modified the training setting for our purpose.

2.1 Pre-processing

As aforementioned, our AMT system is based on the existing model. Therefore, we used the same multi-resolution STFT with semitone spaced logarithmic compression in the model. It first receives audio waveforms as input and computes two types of short time Fourier transform (STFT), one with a short window (2048 samples, 46.4 ms) and the other with a long window (8192 samples, 185.8 ms), with the same overlap (441 samples, 10 ms). The STFT with a short time window gives temporally sensitive output while the one with a longer window offers better frequency resolution. A Hamming window was applied on the signal before the STFT. We only take magnitude of the STFT, thereby obtaining spectrogram with 100 frames/sec.

To apply logarithmic characteristics of sound intensity, a log-like compression with a multiplication factor 1000 is applied on the magnitude of spectrograms. We then reduce the dimensionality of inputs by filtering with semi-tone filterbanks. The center frequencies are distributed according to the frequencies of the 88 MIDI notes and the widths are formed with overlapping triangular shape. This process is not only effective for reducing size of inputs but also for suppressing variance in piano tuning by merging neighboring frequency bins. In the low frequency, some note bins become completely zero or linear summation of neighboring notes due to the low frequency resolution of the spectrogram. We remove those dummy note bins, thereby having 183 dimensions in total. We augmented the input by concatenating it with the first-order difference of the semitone filtered spectrogram. We observed a significant increase of the transcription performance with this addition.

2.2 Neural Network

The Böck and Schedl model uses a recurrent neural network (RNN) using the Long Short Term Memory (LSTM) architecture. Compared to feedforward neural networks, RNNs are capable of learning temporal dependency of sequential data, which is the property found in music audio. Also, the LSTM unit has a memory block updated only when an input or forget gate is open, and the gradients can propagate through memory cells without being multiplied each time step. This property enables LSTM to learn long-term dependency. In our task, the LSTM is expected to learn the continuity of onset, sustain and offset within a note as well as the relation among notes. The LSTM units are also set to be bidirectional, indicating that the input sequence is not only presented in order but also in the opposite direction. Throughout backward and forward layers together, the networks can access to both history and future of the given time frame.

While the Böck and Schedl model used a single network that predicts 88 notes, we use two types of networks; one predicts 88 notes or 12 chroma and the other predicts 12 chroma onsets. In the 88-note network, we reduced the size to two layers of 200 LSTM units as it performed better in our experiments. In the 12-chroma, we downsized it further having 100 LSTM units on the first layer and 50 LSTM units on the second layer. On top of the LSTM networks, a fully connected layer with sigmoid activation units are added as the output layer. Each output unit corresponds to one MIDI note or chroma (i.e. pitch class of the MIDI note).

2.2.1 *Backpropagation*

Theoretically, LSTM can learn any length of long-term dependency through backpropagation through time (BPTT) with a desired number of time steps. In practice, it requires large memory and heavy computation because all past history of network within the backpropagation length should be stored and updated. To overcome this difficulty, a truncated backpropagation method [9] is usually applied for long sequences (also with long time dependency). In the truncated backpropagation, input sequences are divided into shorter sequences and the last state of each segment is transfered to the consecutive segment. Therefore, even though the backpropagation is only computed in each segment, it can serve as an approximation to full-length backpropagation. For a bidirectional system, however, the backward flow requires computation on the full future and thus the truncated backpropagation requires large memory as well. To imitate the advantage of the truncated backpropagation within the computational availability, we split the input sequence into relatively large sequences and perform full-length backpropagation within each segment. We conducted grid search on the segmentation length between 10

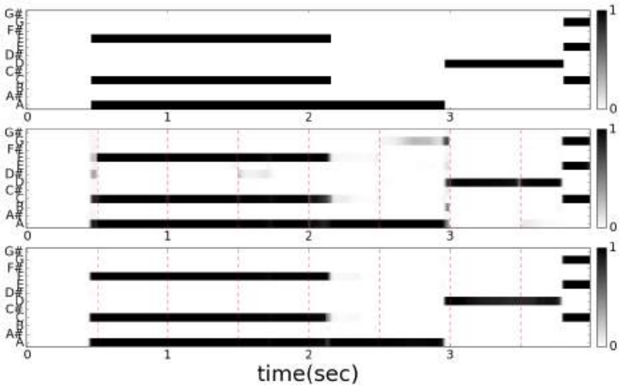


Figure 2. Examples of bidirectional LSTM networks that predict 12 chroma: (a) ground truth, (b) without overlapping segmentation, (c) with overlapping segmentation. Dotted lines indicate the boundaries of segments

frames to 300 frames (100 to 3000 ms) and finally settled down to 50 frames (500 ms). This was long enough to catch up the continuity of individual notes and also was not computationally expensive. We conducted a comparative experiment between a unidirectional model with the truncated backpropagation and a bidirectional model with a non-transferred segmentation. The result showed that the bidirectional model performs better.

To reduce the amount computation, our model works in sequence-to-sequence manner. In other words, the output of the network is a sequence with the same length of input segment. Therefore, frames on the edges of a segment have only one-side context window. We observed that contagious errors frequently occur on such frames as shown in Figure 2b. To tackle this problem we split the input sequence with 50% overlapped segments and take only the middle part of output from each segment. This procedure significantly increase transcription result as shown in Figure 2c.

2.2.2 *Network Training*

In order to train the networks, we used audio files and aligned MIDI files. The MIDI data was converted into a piano-roll representation with the same frame rate of the input filter-bank spectrogram (100 fps). For 88 notes and chroma labels, the elements of piano-roll representation were set to 1 between note onset and offset and otherwise to 0. For chroma onset labels, only the elements that correspond to note onsets were set to 1. The corresponding audio data was normalized with zero-mean and standard deviation of one over each filter in the training set.

We used dropout with a ratio of 0.5 and weight regularization with a value of 10^{-4} in each LSTM layer. This effectively improved the performance by generalization. We used the network with stochastic gradient decent to minimize binary cross entropy loss function. Learning rate was initially set as 0.1 and iteratively decreased by a factor of 3 when no improvement was observed for validation loss for 10 epochs (i.e. early stopping). The training was stopped after six iterations. Examples of the AMT outputs are presented in Figure 3. To verify the performance, frame-wise transcription performance for the 88-note AMT system was measured on the test sets. We used a fixed threshold of 0.5 to predict the note presence and measured the accuracy with F-score to make the results comparable to those in [10, 11]. The resulting F-score was 0.7285 on average, which is better than the results of RNN with basic units [11] and lower than those with fine-tuned frame-wise DNN and CNN [10].

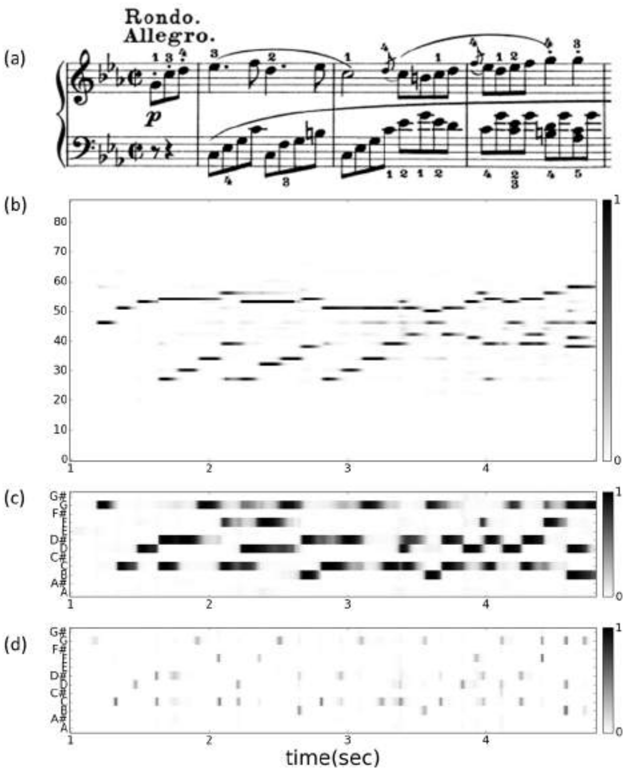


Figure 3. (a) An excerpt of music score from Beethoven's 8th sonata. (b)-(d) the prediction outputs of the AMT systems: (b) 88 note (c) 12 chroma (d) 12 chroma onset.

2.3 Alignment

The AMT systems return two types of MIDI-level features. For chroma onset features, every onset was elongated for 10 frames (100 ms) with decaying weights of 1, $\sqrt{0.9}$, $\sqrt{0.8}$, ... , $\sqrt{0.1}$ as proposed in [6]. The resulting features are combined by concatenation with either 88-note or 12-chroma AMT output features. The corresponding score MIDI was also converted into 88 note (or chroma) and the chroma onsets are elongated in the same manner before combined.

We used euclidean distance to measure similarity between the two combined representations. We then applied the FastDTW algorithm [12] which is an approximate method to dynamic time warping (DTW). FastDTW uses iterative multi-level approach with window constraints to reduce the complexity. Because of the high frame rate of the features, it is necessary to employ low-cost algorithm. While the original DTW algorithm has O(N^2) time and space complexity, FastDTW operates in O(N) complexity with almost the same accuracy. Müller et al. [13] also examined a similar multi-level DTW for the audio-to-score alignment task and reported similar results compared to the original DTW. The radius parameter in the fastDTW algorithm, which defines the size of window to find an optimal path for each resolution refinement, was set to 10 in our experiment.

3. EXPERIMENTS

3.1 Dataset

We used the MAPS dataset [14], specifically the 'MUS' subset that contains large pieces of piano music, for training and evaluation. Each piece consists of audio files and a ground-truth MIDI file. The audio files were rendered from the MIDI with nine settings of different pianos and recording conditions. This helped our model avoid overfitting to a specific piano tone. The MIDI files served as the ground-truth annotation of the corresponding audio but some of them (ENSTDkCl and ENSTDkAm) are sometimes temporally inaccurate, which is more than 65 ms as described in [15].

We conducted the experience with 4-fold cross validation with training and test splits from the configuration I [1] in [11]. For each fold, 43 pieces were detached from the training set and used for validation. As a result, each fold was composed of 173, 43 and 54 pieces for training, validation and test, respectively, as processed in [10].

3.2 Evaluation method

In order to evaluate the audio-to-score alignment task, we need another MIDI representation (typically score MIDI) apart from the performance MIDI aligned with audio. We generated the separate MIDI by changing intervals between successive concurrent set of notes. Specifically, we multiplied a randomly selected value between 0.7 and 1.3 to modify the interval. This scheme of temporal distortion prevents the alignment path from being trivial and was also employed in previous work [6, 16, 17].

After we obtained the alignment path through DTW, absolute temporal errors between estimated note onsets and ground truth were measured. For each piece of music in the test set, mean value of the temporal errors and ratio of correctly aligned notes with varying thresholds were used to summarize the results.

3.3 Compared Algorithms

To make a performance comparison, we reproduced two alignment algorithms proposed by Ewert et al. [6] and one by Carabias-Orti et al. [18]. We performed the experiments with the same test set using the FastDTW algorithm but without any post-processing. Ewert's algorithms used a hand-crafted chromagram and onset features based on audio filter bank responses. Carabias-Orti's algorithm employed a non-negative matrix factorization to learn spectral basis of each note combination from spectrogram. The latter is designed only for audio-to-audio alignment while the former can be applied to both audio-to-audio and audio-to-MIDI alignment. Therefore, we made an audio version of the distorted MIDI using a high-quality sample-based piano synthesizer and employed it as an input. We tested Ewert's algorithms for both audio and MIDI cases. The temporal frame rate of features were adjusted to 100 fps for both algorithms.

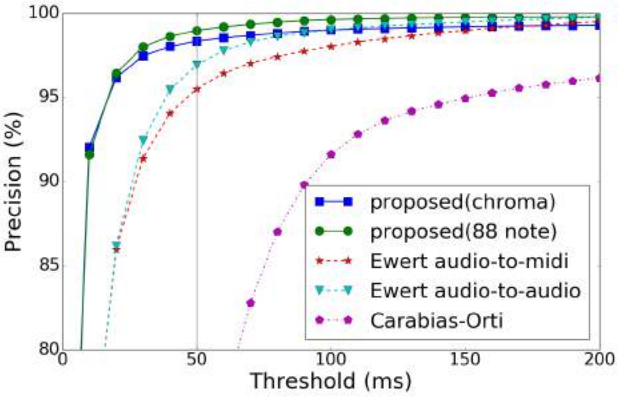


Figure 4. Ratio of correctly aligned onsets as a function of threshold. Each point represent mean of piecewise precision. Some data points with lower than 80% of precision are not shown in this figure.

For the aligning task with Ewert's algorithms, we used the same FastDTW algorithm. But since the FastDTW algorithm cannot be directly applied to Carabias-Orti's algorithm due to its own distance calculation method, we applied a classic DTW algorithm, which employs an entire frame-wise distance matrix. Because of the limitation of memory, when reproducing Carabias-Orti's algorithm, we excluded 35 pieces that are longer than 400 seconds among the test sets.

Note that even though the dataset for evaluation is different, the results of two reproduced algorithms were similar to the results in their original works. The mean onset errors of Ewert's algorithm on piano music was 19 ms with 26 ms standard deviation [6]. The result introduced in the original Carabias-Orti's paper [18] shows a quite large difference in terms of the mean of piecewise error, but we assumes that the difference is due to the change of the test set. The align rate of original result and our reproduced result were similar (50 ms: 74% - 69%, 100 ms: 90% - 92%, 200 ms: 95% - 96%). Hence, we assumed that our reproduction was reliable for the comparison.

4. RESULTS AND DISCUSSION

4.1 Comparison with Others

Figure 4 shows the results of the audio-to-score alignment from the compared algorithms. They represent the ratio of correctly aligned onsets in precision as a function of error threshold. Typically, tolerance window with 50 ms is used for evaluation. However, because most of notes were aligned within 50 ms of temporal threshold, we varied the width tolerance window from 0 ms to 200 ms with 10 ms steps.

[1] http://www.eecs.qmul.ac.uk/sss31/TASLP/info.html

		Mean	Median	Std	≤ 10 ms	≤ 30 ms	≤ 50 ms	≤ 100 ms
Proposed with onset	chroma	12.83	6.40	56.22	**92.01**	97.44	98.31	98.98
	88 note	**8.62**	**5.57**	31.14	91.60	**98.00**	**98.97**	**99.61**
Proposed w/o onset	chroma	48.01	27.96	152.06	60.66	84.65	89.36	93.72
	88 note	25.31	18.69	63.26	56.39	86.42	93.05	97.48
Ewert et. al. (audio-to-MIDI)		16.44	13.64	32.52	71.78	91.38	95.50	98.03
Ewert et. al. (audio-to-audio)		14.66	11.71	**25.38**	71.53	92.43	96.91	99.13
Carabias-Orti et. al.		131.31	49.96	305.52	23.58	49.40	69.30	91.60

Table 1. Results of the piecewise onset errors. Mean, median, and standard deviation of the errors are in millisecond. The right columns are the ratio of notes (%) that are aligned within the onset error of 10 ms, 30 ms, 50 ms and 100 ms, respectively.

Overall, our proposed framework with 88 note combined with the chroma onsets achieved the best accuracy. Even with zero threshold, which means the best match with resolution of our system (10 ms), our proposed model with the 88-note output exactly aligned 52.55% of notes. The ratio was increased to 91.60% with 10 ms threshold. The proposed framework using 12 chroma showed similar precision to the 88-note framework, but the accuracy was slightly lower. Compared to Ewert's algorithms with hand-crafted features, our method shows significantly better performance especially in high resolution. Over 100 ms of threshold, our framework with chroma and Ewert's method shows similar precisions but the difference becomes significant with the intervals under 50 ms. Note that we penalized our framework compared to the audio-to-audio scenario of Ewert's algorithm because the audio-to-audio approach takes advantage from identical note velocities. We suppose Ewert's algorithm performed better in the audio-to-audio scenario rather than the audio-to-MIDI for the same reason. Carabias-Orti's algorithm shows lower precisions compared to others. We assume that the difference mainly comes from the usage of onset features.

For the fair comparison of the results, we should note that our framework is heavily dependent on the training set unlike the two other compared methods. On the other hand, Carabias-Orti's algorithm focused on dealing with various instruments and online alignment scenario, which were unable to be fully appreciated in our experiment.

4.2 Effect of Chroma Onset Features

On the second experiment, we further investigate the effect of chroma onset features. We removed the onset features from each model and compared the mean onset errors and shows their distribution. As can be seen in Figure 5, the absence of onset features significantly decreases the performance. Thus we conclude that employing chroma onset features can compensate the limitation of normalized transcription features. As we stated in Section 1, 88 note representation shows much better results compared to those with chroma output features especially without onset.

Table 1 shows the statistics of piecewise onset errors. This shows that the use of chroma onset feature is crucial in our proposed method. The median of piecewise onset errors was decreased from 18.69 ms to 5.57 ms when applying chroma onset features to the 88-note system. The importance of the note onset feature for aligning piano music was also examined in [6].

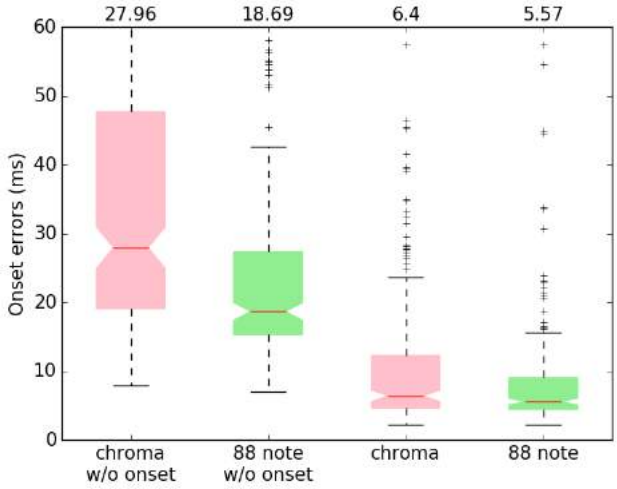


Figure 5. Comparison of mean onset errors between models with/without chroma onset features. Each point corresponds to mean onset error of a piece. Outliers above 60 ms of errors are omitted in this figure. Each number on the top of the box indicates the median value in ms.

In addition to these experiments, we also aligned some real-world recordings through the trained system. Even though a quantitative evaluation has not been presented here, the sonification of the aligned MIDI files shows promising results. The synchronized MIDI-audio examples are linked on our demo website [2].

5. CONCLUSIONS

In this paper, we proposed a framework for audio-to-score alignment of piano music using automatic music transcription. We built two AMT systems based on bidirectional LSTM that predict note existence and chroma onset. They provide MIDI-level features that can be compared with score MIDI to be used for the alignment algorithm. Our experiments with the MAPS dataset showed that the AMT-based features are effective in the alignment task and our proposed system outperforms compared approaches. The 88-note model with chroma onset worked best. We also

[2] http://mac.kaist.ac.kr/~ilcobo2/alignWithAMT

showed that chroma onset features take a crucial role in improving the accuracy. In fact, the successful alignment performance might be possible because we used the same recording condition for both training and test sets. Considering this issue, we investigate the generalization capacity of our model by evaluating it on various datasets in the future. Also, we plan to improve the AMT system by using other types of deep neural networks.

Acknowledgments

This work was supported by Korea Advanced Institute of Science and Technology (project no. G04140049) and Korea Creative Content Agency (project no. N04170044).

6. REFERENCES

[1] A. Arzt, G. Widmer, and S. Dixon, "Automatic Page Turning for Musicians via Real-Time Machine Listening," in *Proceedings of the conference on ECAI 2008: 18th European Conference on Artificial Intelligence*, vol. 1, no. 1, 2008, pp. 241–245.

[2] R. B. Dannenberg and C. Raphael, "Music score alignment and computer accompaniment," *Communications of the ACM*, vol. 49, no. 8, pp. 38–43, 2006.

[3] G. Widmer, S. Dixon, W. Goebl, E. Pampalk, and A. Tobudic, "In Search of the Horowitz Factor," *AI Magazine*, vol. 24, no. 3, pp. 111–130, 2003.

[4] A. Friberg and J. Sundberg, "Perception of just-noticeable time displacement of a tone presented in a metrical sequence at different tempos," *The Journal of The Acoustical Society of America*, vol. 94, no. 3, pp. 1859–1859, 1993.

[5] S. Dixon and G. Widmer, "MATCH: a music alignment tool chest," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2005, pp. 492–497.

[6] S. Ewert, M. Müller, and P. Grosche, "High resolution audio synchronization using chroma onset features," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2009, pp. 1869–1872.

[7] A. Arzt, S. Böck, S. Flossmann, H. Frostel, M. Gasser, C. C. S. Liem, and G. Widmer, "The piano music companion," *Frontiers in Artificial Intelligence and Applications*, vol. 263, no. 1, pp. 1221–1222, 2014.

[8] S. Böck and M. Schedl, "Polyphonic piano note transcription with recurrent neural networks," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2012, pp. 121–124.

[9] R. J. Williams and J. Peng, "An Efficient Gradient-Based Algorithm for On-Line Training of Recurrent Network Trajectories," *Appears in Neural Computation*, no. 2, pp. 490–501, 1990.

[10] R. Kelz, M. Dorfer, F. Korzeniowski, S. Böck, A. Arzt, and G. Widmer, "On the Potential of Simple Framewise Approaches to Piano Transcription," in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2016, pp. 475–481.

[11] S. Sigtia, E. Benetos, and S. Dixon, "An end-to-end neural network for polyphonic piano music transcription," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 24, no. 5, pp. 927–939, 2016.

[12] S. Salvador and P. Chan, "FastDTW : Toward Accurate Dynamic Time Warping in Linear Time and Space," *Intelligent Data Analysis*, vol. 11, pp. 561–580, 2007.

[13] M. Müller, H. Mattes, and F. Kurth, "An efficient multiscale approach to audio synchronization," in *Proc. International Conference on Music Informa- tion Retrieval (ISMIR)*, 2006, p. 192197.

[14] V. Emiya, R. Badeau, and B. David, "Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 18, no. 6, pp. 1643–1654, 2010.

[15] S. Ewert and M. Sandler, "Piano Transcription in the Studio Using an Extensible Alternating Directions Framework," vol. 24, no. 11, pp. 1983–1997, 2016.

[16] M. Müller, H. Mattes, and F. Kurth, "An efficient multiscale approach to audio synchronization." in *Proc. International Conference on Music Informa- tion Retrieval (ISMIR)*. Citeseer, 2006, pp. 192–197.

[17] C. Joder, S. Essid, and G. Richard, "A conditional random field framework for robust and scalable audio-to-score matching," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 8, pp. 2385–2397, 2011.

[18] J. J. Carabias-Orti, F. J. Rodríguez-Serrano, P. Vera-Candeas, N. Ruiz-Reyes, and F. J. Cañadas-Quesada, "An audio to score alignment framework using spectral factorization and dynamic time warping." in *Proc. International Conference on Music Informa- tion Retrieval (ISMIR)*, 2015, pp. 742–748.

L^p-VITERBI ALGORITHM FOR AUTOMATIC FINGERING DECISION

Gen Hori
Asia University / RIKEN
hori@brain.riken.jp

Shigeki Sagayama
Meiji University
sagayama@meiji.ac.jp

ABSTRACT

A theoretical basis of existing attempts to automatic fingering decision is path optimization that minimizes the difficulty of whole phrase that is typically defined as the sum of the difficulties of each moves required for playing the phrase. However, from a practical point of view, it is more important to minimize the maximum difficulty of the move required for playing the phrase, that is, to make the most difficult move easiest. For this reason, our previous work introduced a variant of the Viterbi algorithm termed the "minimax Viterbi algorithm" that finds the path of the hidden states that maximizes the minimum transition probability along the path. In the present work, we introduce a parameterized family of Viterbi algorithm termed the "L^p-Viterbi algorithm" that continuously interpolates the conventional Viterbi algorithm and the minimax Viterbi algorithm. (It coincides with the conventional for $p = 1$ and the minimax for $p = \infty$.) We apply those variants of the Viterbi algorithm to HMM-based guitar fingering decision and compare the resulting fingerings.

1. INTRODUCTION

The pitch ranges of strings of string instruments usually have significant overlaps. Therefore, such string instruments have several ways to play even a single note (except the highest and the lowest notes that are covered only by a single string) and thus numerous ways to play a phrase and an astronomical number of possible ways to play a whole song. This is why fingering decision for a given song is not an easy task and automatic fingering decision has been attracting many researchers. Existing attempts to automatic fingering decision are mainly based on path optimization by minimizing the difficulty level of whole phrase that is defined as the sum or the product of the difficulty levels of each moves. However, whether a string player can play a given passage using a specific fingering depends almost only on whether the most difficult move included in the fingering is playable. In particular, for beginner players, it is most important to minimize the maximum difficulty level of move included in a fingering, that is, to "make the most difficult move easiest."

To this end, our previous work [1] introduced a variant of the Viterbi algorithm [2] termed the "minimax Viterbi algorithm" for finding the sequence of the hidden states that maximizes the minimum transition probability on the sequence (not the product of all the transition probabilities on the sequence). The purpose of the paper is to introduce a parameterized family of Viterbi algorithm termed the "L^p-Viterbi algorithm" that continuously interpolates the conventional Viterbi algorithm and the minimax Viterbi algorithm and then apply it to HMM-based guitar fingering decision. The framework of HMM-based fingering decision employs a hidden Markov model (HMM) whose hidden states are left hand forms of guitarists and output symbols are musical notes. We perform fingering decision by solving the decoding problem of HMM using our proposed L^p-Viterbi algorithm. We set the transition probabilities to large values for easy moves and small values for difficult ones so that resulting fingerings make the most difficult move easiest as previously discussed in this section. To distinguish the original Viterbi algorithm from our variants, we refer to the former as "conventional Viterbi algorithm" throughout the paper.

There have been several attempts to automatic fingering decision and automatic arrangement for guitars. Sayegh [3] introduced the path optimization approach to fingering decision of generic string instruments. Miura et al. [4] developed a system that generates guitar fingerings for given melodies (monophonic phrases). Radicioni et al. [5] introduced cognitive aspects of fingering decision to the path optimization approach. Radisavljevic and Driessen [6] proposed a method for designing cost functions for dynamic programming (DP) for fingering decision. Tuohy and Potter [7] introduced a genetic algorithm (GA) for fingering decision. Baccherini et al. [8] introduced finite state automaton to fingering decision of generic string instruments. Hori et al. [9] applied input-output HMM introduced by Bengio and Frasconi [10] to guitar fingering decision and arrangement. McVicar et al. [11] developed a system that generates guitar tablatures for rhythm guitar and lead guitar automatically. Comparing to those previous works, the novelty of the present work lies in that it introduces a new category of path optimization and variants of the Viterbi algorithm correspondingly.

The rest of the paper is organized as follows. Section 2 recalls the conventional Viterbi algorithm and the minimax Viterbi algorithm and connects them by introducing the L^p-Viterbi algorithm. Section 3 recalls the framework of fingering decision based on HMM for monophonic guitar phrases to which we apply the L^p-Viterbi algorithm and evaluates the resulting fingerings in Section 4. Section 5 concludes the paper.

2. L^P-VITERBI ALGORITHM

First of all, we recall the definition of HMM and the procedure of the conventional Viterbi algorithm for finding the sequence of hidden states with the maximum likelihood. Next, we look at the minimax Viterbi algorithm introduced in our previous work [1] for finding the sequence of hidden states with the maximum minimum transition probability. After that, we introduce the L^p-Viterbi algorithm that continuously interpolates the conventional Viterbi algorithm and the minimax Viterbi algorithm.

2.1 Hidden Markov model (HMM)

Suppose that we have two finite sets of hidden states Q and output symbols S,

$$Q = \{q_1, q_2, \dots, q_N\},$$
$$S = \{s_1, s_2, \dots, s_M\},$$

and two sequences of random variables $\boldsymbol{X}$ of hidden states and $\boldsymbol{Y}$ of output symbols,

$$\boldsymbol{X} = (X_1, X_2, \dots, X_T), \quad X_t \in Q,$$
$$\boldsymbol{Y} = (Y_1, Y_2, \dots, Y_T), \quad Y_t \in S,$$

then a hidden Markov model M is defined by a triplet

$$M = (A, B, \pi)$$

where A is an $N \times N$ matrix of the transition probabilities,

$$A = (a_{ij}),\ a_{ij} \equiv a(q_i, q_j) \equiv P(X_t = q_j | X_{t-1} = q_i),$$

B an $N \times K$ matrix of the output probabilities,

$$B = (b_{ik}),\ b_{ik} \equiv b(q_i, s_k) \equiv P(Y_t = s_k | X_t = q_i),$$

and Π an N-dimensional vector of the initial distribution of hidden states,

$$\Pi = (\pi_i),\ \pi_i \equiv \pi(q_i) \equiv P(X_1 = q_i).$$

2.2 Conventional Viterbi algorithm

When we observe a sequence of output symbols

$$\boldsymbol{y} = (y_1, y_2, \dots, y_T)$$

from a hidden Markov model M, we are interested in the sequence of hidden states

$$\boldsymbol{x} = (x_1, x_2, \dots, x_T)$$

that generated the observed sequence of output symbols $\boldsymbol{y}$ with the maximum likelihood,

$$\begin{aligned}\hat{\boldsymbol{x}}_{ML} &= \arg\max_{\boldsymbol{x}} P(\boldsymbol{y}, \boldsymbol{x} | M) \\ &= \arg\max_{\boldsymbol{x}} P(\boldsymbol{x}|M) P(\boldsymbol{y}|\boldsymbol{x}, M) \\ &= \arg\max_{\boldsymbol{x}} (\log P(\boldsymbol{x}|M) + \log P(\boldsymbol{y}|\boldsymbol{x}, M)) \\ &= \arg\max_{\boldsymbol{x}} \sum_{t=1}^{T} (\log a(x_{t-1}, x_t) + \log b(x_t, y_t))\end{aligned} \tag{1}$$

where we write $\pi(x_1) = a(x_0, x_1)$ for convenience. The problem of finding the maximum likelihood sequence $\hat{\boldsymbol{x}}_{ML}$ is called the decoding problem and solved efficiently using two $N \times T$ tables $\Delta = (\delta_{it})$ of maximum log likelihood and $\Psi = (\psi_{it})$ of back pointers and the following four steps.

Initialization initializes the first columns of the two tables Δ and Ψ using the following formulae for $i = 1, 2, \dots, N$,

$$\delta_{i1} = \log \pi_i + \log b(q_i, y_1),$$
$$\psi_{i1} = 0.$$

Recursion fills out the rest columns of Δ and Ψ using the following recursive formulae for $j = 1, 2, \dots, N$ and $t = 1, 2, \dots, T$–1,

$$\delta_{j,t+1} = \max_i(\delta_{it} + \log a_{ij}) + \log b(q_j, y_{t+1}),$$
$$\psi_{j,t+1} = \arg\max_i(\delta_{it} + \log a_{ij}).$$

Termination finds the last hidden state of the maximum likelihood sequence $\hat{\boldsymbol{x}}_{ML}$ using the last column of Δ,

$$x_T = \arg\max_i \delta_{iT}.$$

Backtracking tracks the maximum likelihood sequence $\hat{\boldsymbol{x}}_{ML}$ from the last to the first using the back pointers of Ψ for $t = T, T$–1$, \dots, 2$,

$$x_{t-1} = \psi_{x_t, t}.$$

2.3 Minimax Viterbi algorithm

To implement a variant of the conventional Viterbi algorithm for finding the sequence of hidden states $\hat{\boldsymbol{x}}_{MM}$ with the maximum minimum transition probability,

$$\hat{\boldsymbol{x}}_{MM} = \arg\max_{\boldsymbol{x}}\ \min_t(\log a(x_{t-1}, x_t) + \log b(x_t, y_t)) \tag{2}$$

where the minimum value is taken from $t = 1, 2, \dots, T$, we modify the second step of the conventional Viterbi algorithm as follows.

Recursion for minimax Viterbi algorithm fills out the two tables Δ and Ψ using the following recursive formulae for $j = 1, 2, \dots, N$ and $t = 1, 2, \dots, T$–1,

$$\delta_{j,t+1} = \max_i(\min(\delta_{it},\ \log a_{ij} + \log b(q_j, y_{t+1}))),$$
$$\psi_{j,t+1} = \arg\max_i(\min(\delta_{it},\ \log a_{ij} + \log b(q_j, y_{t+1}))).$$

We modify only the second step and leave other steps unchanged. We call the modified algorithm the "minimax Viterbi algorithm." The modified algorithm finds the sequence of hidden states with the maximum minimum probability, that is, the minimum maximum difficulty. Although the word "maximin" is appropriate in view of probability, we choose the word "minimax" for naming the variant because it is intuitive to understand in view of difficulty and the word conveys our concept of "making the most difficult move easiest." The modified second step works in the same manner as the original one but now the element δ_{it} keeps the value of the maximum minimum transition probability of the subsequence of hidden states for the first t observations. The term $\min(\delta_{it},\, \log a_{ij} + \log b(q_j, y_{t+1}))$ updates the value of the minimum transition probability as the term $\delta_{it} + \log a_{ij} + \log b(q_j, y_{t+1})$ does the value of the maximum log likelihood in the conventional Viterbi algorithm.

2.4 L^p-Viterbi algorithm

To implement a family of Viterbi algorithm that continuously interpolates the conventional Viterbi algorithm and the minimax Viterbi algorithm, we consider a generalized decoding problem of HMM defined with the L^p-norm of a certain real vector. For a real number $p \geq 1$ [1], the L^p-norm of an n-dimensional real vector

$$\boldsymbol{v} = (v_1, v_2, \ldots, v_n) \in R^n$$

is defined as

$$||\boldsymbol{v}||_p = \sqrt[p]{|v_1|^p + |v_2|^p + \cdots + |v_n|^p}. \qquad (3)$$

In particular, the L^1-norm and the L^2-norm correspond to the Manhattan distance and the Euclidean distance and are widely used in the contexts of machine learning and compressed sensing. The regression models with the L^1 and L^2 regularizations are the LASSO regression and the ridge regression respectively. The L^∞-norm (the limit of the L^p-norms for $p{\to}\infty$) is equivalent to the following definition,

$$||\boldsymbol{v}||_\infty = \max\{|v_1|, |v_2|, \ldots, |v_n|\}.$$

We define two T-dimensional real vectors of the transition probabilities $\boldsymbol{a}$ and the output probabilities $\boldsymbol{b}$ along the sequences of hidden states $\boldsymbol{x}$ and output symbols $\boldsymbol{y}$,

$$\begin{aligned}\boldsymbol{a}(\boldsymbol{x}) &= (\pi(x_1), a(x_1, x_2), \ldots, a(x_{T-1}, x_T)),\\ \boldsymbol{b}(\boldsymbol{x}, \boldsymbol{y}) &= (b(x_1, y_1), b(x_2, y_2), \ldots, b(x_T, y_T)),\end{aligned}$$

and consider a generalized decoding problem,

$$\hat{\boldsymbol{x}}_{L^p} = \arg\min_{\boldsymbol{x}} ||\log \boldsymbol{a}(\boldsymbol{x}) + \log \boldsymbol{b}(\boldsymbol{x}, \boldsymbol{y})||_p, \qquad (4)$$

where log operates element-wise on vectors. The special cases of $p{=}1$ and $p{=}\infty$ of the generalized decoding problem (4) coincide with the conventional decoding problem (1) and the minimax decoding problem (2) respectively. Thus the generalized decoding problem continuously interpolates the conventional and the minimax decoding problems. Note that $\arg\max$ in (1) and (2) changes to $\arg\min$ in (4) because the logarithms of probabilities are always negative or zero and therefore taking the absolute values of them in the L^p-norm (3) always inverts their signs. All the elements of the table Δ are negative or zero in the conventional and the minimax Viterbi algorithm while they are all positive or zero in the generalized decoding problem. We call the generalized decoding problem (4) the "L^p-decoding problem." To solve the L^p-decoding problem efficiently, we modify the second step of the conventional Viterbi algorithm as follows. According to the change of signs in the table Δ, we also modify the first and third step as follows. We leave the last step unchanged.

Initialization for L^p-Viterbi algorithm initializes the first columns of the two tables Δ and Ψ using the following formulae for $i = 1, 2, \ldots, N$,

$$\begin{aligned}\delta_{i1} &= |\log \pi_i + \log b(q_i, y_1)|,\\ \psi_{i1} &= 0.\end{aligned}$$

Recursion for L^p-Viterbi algorithm fills out the two tables Δ and Ψ using the following recursive formulae for $j = 1, 2, \ldots, N$ and $t = 1, 2, \ldots, T$–1,

$$\begin{aligned}\delta_{j,t+1} &= \min_i \sqrt[p]{\delta_{it}{}^p + |\log a_{ij} + \log b(q_j, y_{t+1})|^p},\\ \psi_{j,t+1} &= \arg\min_i \sqrt[p]{\delta_{it}{}^p + |\log a_{ij} + \log b(q_j, y_{t+1})|^p}.\end{aligned}$$

Termination for L^p-Viterbi algorithm finds the last hidden state of the maximum likelihood sequence $\hat{\boldsymbol{x}}_{ML}$ using the last column of Δ,

$$x_T = \arg\min_i \delta_{iT}.$$

3. FINGERING DECISION BASED ON HMM

We evaluate our proposed L^p-Viterbi algorithm in the following section with the same fingering decision model for monophonic guitar phrases used in our previous work [1]. In this section, we look at the monophonic fingering decision model based on HMM whose hidden states are left hand forms and output symbols are musical notes played by the left hand forms. In this formulation, fingering decision is cast as a decoding problem of HMM where a fingering is obtained as a sequence of hidden states.

3.1 HMM for monophonic fingering decision

To play a single note with a guitar, a guitarist depresses a string on a fret with a finger of the left hand and picks the same string with the right hand. Therefore, a form q_i for playing a single note can be expressed in a triplet

$$q_i = (s_i, f_i, h_i)$$

where $s_i = 1, \ldots, 6$ is a string number (from the highest to the lowest), $f_i = 0, 1, \ldots$ is a fret number, and $h_i = 1, 2, 3, 4$ is a finger number of the player's left hand (1,2,3 and 4 means the index, middle, ring and pinky fingers). The fret number $f_i = 0$ means an open string. For the standard tuning (E_4-B_3-G_3-D_3-A_2-E_2), the MIDI note numbers of the open strings are $o_1 = 64, o_2 = 59$, $o_3 = 55, o_4 = 50, o_5 = 45, o_6 = 40$ from which the MIDI note number of the note played by the form q_i is calculated as

$$n(q_i) = o_{s_i} + f_i.$$

[1] For $p < 1$, $||\boldsymbol{v}||_p$ does not meet the axioms of norm.

3.2 Transition and output probabilities

The model parameters of HMM such as the transition probabilities and the output probabilities are usually estimated from training data using the Baum-Welch algorithm [12]. However, we set those parameters manually as explained in the following instead of estimation from training data because it is difficult to prepare enough training data for our application of fingering decision, that is, guitar scores annotated with fingerings.

The difficulty levels of the moves from forms to forms are implemented in the probabilities of the transitions from hidden states to hidden states; a small value of the transition probability means the corresponding move is difficult and a large value means easy. For simplicity, we assume that the four fingers of the left hand (the index, middle, ring and pinky fingers) are always put on consecutive frets. This lets us calculate the *index finger position* (the fret number the index finger is put on) of form q_i as follows,

$$g(q_i) = f_i - h_i + 1.$$

Using the index finger position, we set the transition probability as

$$\begin{aligned} a_{ij}(d_t) &= P(X_t = q_j \mid X_{t-1} = q_i,\, d_t) \\ &\propto \frac{1}{2d_t} \exp\left(-\frac{|g(q_i) - g(q_j)|}{d_t}\right) \times P_H(h_j) \end{aligned}$$

where $\propto$ means proportional and the left hand side is normalized so that the summation with respect to j equals 1. The first term of the right hand side is taken from the probability density function of the Laplace distribution that concentrates on the center and its variance d_t is set to the time interval between the onsets of the $(t-1)$-th note and the t-th note. The second term $P_H(h_j)$ corresponds to the difficulty level of the destination form defined depending on the finger number h_j. In the simulation in the following section, we set $P_H(1) = 0.4$, $P_H(2) = 0.3$, $P_H(3) = 0.2$ and $P_H(4) = 0.1$ which means the form using the index finger is the easiest and the pinky finger is the most difficult. The difficulty levels of the forms are included in the transition probabilities (not in the output probability) in such a way that a transition probability is adjusted to a smaller value when the destination form of the transition is difficult.

As for the output probability, because all the hidden states have unique output symbols in our HMM for fingering decision, it is 1 if the given output symbol is the one that the hidden state outputs and 0 if the given output symbol is not,

$$\begin{aligned} b_{ik} &= P(\,Y_t = s_k \mid X_t = q_i\,) \\ &= \begin{cases} 1 & (\text{if } s_k = n(q_i)) \\ 0 & (\text{if } s_k \neq n(q_i)) \end{cases} . \end{aligned}$$

4. EVALUATION

We evaluate our proposed L^p-Viterbi algorithm by comparing the results of fingering decision for monophonic guitar phrases using the conventional Viterbi algorithm, the minimax Viterbi algorithm, and the proposed algorithm. Figure 1 shows the results for the opening part of “Romance Anonimo.” The numbers on the tablatures show the fret numbers and the numbers in parenthesis below the tablatures show the finger numbers where 1,2,3 and 4 means the index, middle, ring and pinky fingers. In the bottom graph of the transition probabilities, we see that the red line (the conventional Viterbi algorithm) keeps higher values (easy moves) at the cost of two very small values (very difficult moves) while the blue line (the minimax Viterbi algorithm) circumvents such very small values and makes the most difficult move easiest. The green line (the L^p-Viterbi algorithm for $p = 3.0$) falls into an intermediate category. The top tablature uses the pinky finger two times while the middle and the bottom ones avoids it. Figure 2 shows the results for an excerpt from the cello part of “Eine Kleine Nachtmusik.” In the bottom graph, we see again that the red line keeps higher values at the cost of few very small values while the blue line circumvents such very small values.

5. CONCLUSIONS

We have introduced a variant of the conventional Viterbi algorithm termed the L^p-Viterbi algorithm that continuously interpolates the conventional Viterbi algorithm and the minimax Viterbi algorithm. The variant coincides with the conventional Viterbi algorithm for $p = 1$ and the minimax Viterbi algorithm for $p = \infty$. We have applied the variant to HMM-based guitar fingering decision. Our previous work [1] showed that the minimax Viterbi algorithm minimizes the maximum difficulty of the move required for playing a given phrase. In the simulation of the present work, we have compared the fingerings obtained by the conventional Viterbi algorithm, the L^p-Viterbi algorithm and the minimax Viterbi algorithm to see that the L^p-Viterbi algorithm is capable of making fingerings that fall into an intermediate category between the conventional Viterbi algorithm and the minimax Viterbi algorithm. We hope that those variants of the Viterbi algorithm find a wide range of applications in a variety of research fields.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Numbers 26240025, 17H00749.

6. REFERENCES

[1] G. Hori and S. Sagayama, “Minimax viterbi algorithm for HMM-based guitar fingering decision,” in *Proceedings of 17th International Society for Music Information Retrieval (ISMIR2016)*, New York City, U.S.A., 2016, pp. 448–453.

[2] A. J. Viterbi, “Error bounds for convolutional codes and an asymptotically optimum decoding algorithm,” *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, 1967.

[3] S. I. Sayegh, “Fingering for string instruments with

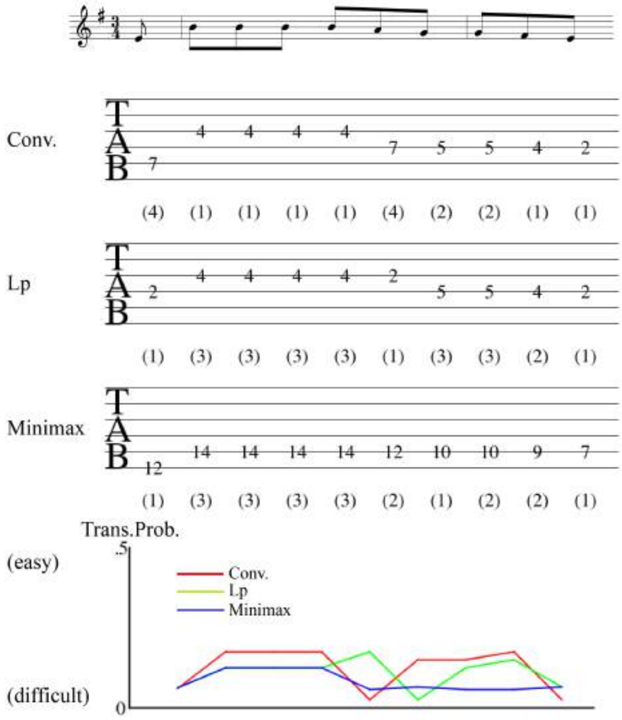


Figure 1. The results of fingering decision for the opening part of "Romance Anonimo" (only top notes). The top, the middle and the bottom tablatures show three fingerings obtained by the conventional Viterbi algorithm, the L^p-Viterbi algorithm ($p = 3.0$) and the minimax Viterbi algorithm respectively. In the bottom graph, the red line shows the transition probabilities along the hidden path obtained by the conventional Viterbi algorithm while the green line and the blue line are obtained by the L^p-Viterbi algorithm and the minimax Viterbi algorithm respectively.

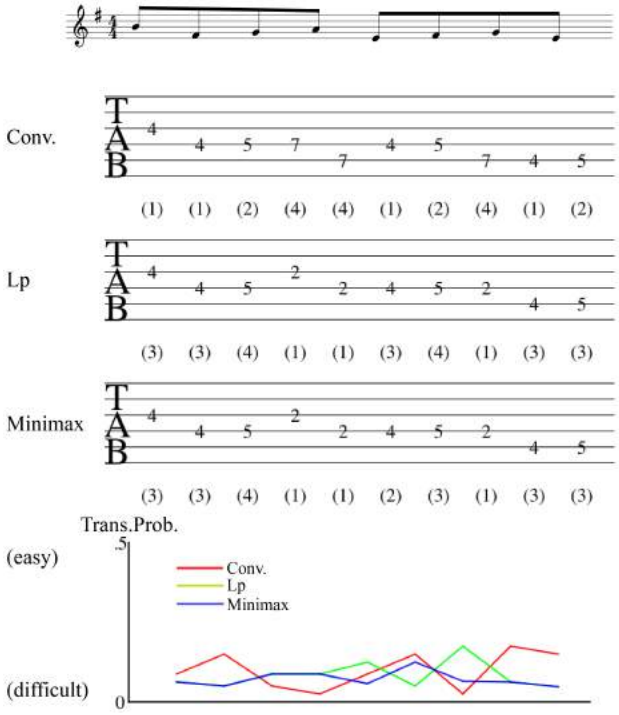


Figure 2. The results of fingering decision for an excerpt from the cello part of "Eine Kleine Nachtmusik." The top, the middle and the bottom tablatures show three fingerings obtained by the conventional Viterbi algorithm, the L^p-Viterbi algorithm ($p = 3.0$) and the minimax Viterbi algorithm respectively. In the bottom graph, the red line, the green line and the blue line are obtained by the conventional Viterbi algorithm, the L^p-Viterbi algorithm and the minimax Viterbi algorithm respectively.

the optimum path paradigm," *Computer Music Journal*, vol. 13, no. 3, pp. 76–84, 1989.

[4] M. Miura, I. Hirota, N. Hama, and M. Yanagida, "Constructing a system for finger-position determination and tablature generation for playing melodies on guitars," *Systems and Computers in Japan*, vol. 35, no. 6, pp. 10–19, 2004.

[5] D. P. Radicioni, L. Anselma, and V. Lombardo, "A segmentation-based prototype to compute string instruments fingering," in *Proceedings of Conference on Interdisciplinary Musicology (CIM04)*, vol. 17, Graz, Austria, 2004, pp. 97–104.

[6] A. Radisavljevic and P. Driessen, "Path difference learning for guitar fingering problem," in *Proceedings of International Computer Music Conference*, vol. 28, Miami, USA, 2004.

[7] D. R. Tuohy and W. D. Potter, "A genetic algorithm for the automatic generation of playable guitar tablature," in *Proceedings of International Computer Music Conference*, 2005, pp. 499–502.

[8] D. Baccherini, D. Merlini, and R. Sprugnoli, "Tablatures for stringed instruments and generating functions," in *Fun with Algorithms*. Springer, 2007, pp. 40–52.

[9] G. Hori, H. Kameoka, and S. Sagayama, "Input-output HMM applied to automatic arrangement for guitars," *Journal of Information Processing*, vol. 21, no. 2, pp. 264–271, 2013.

[10] Y. Bengio and P. Frasconi, "An input output HMM architecture," *Advances in neural information processing systems*, vol. 7, pp. 427–434, 1995.

[11] M. McVicar, S. Fukayama, and M. Goto, "Autoleadguitar: Automatic generation of guitar solo phrases in the tablature space," in *12th International Conference on Signal Processing (ICSP)*, 2014, pp. 599–604.

[12] L. E. Baum and T. Petrie, "Statistical inference for probabilistic functions of finite state Markov chains," *The annals of mathematical statistics*, vol. 37, no. 6, pp. 1554–1563, 1966.

Systema Naturae: shared practices between physical computing and algorithmic composition

Andrea Valle
CIRMA-StudiUm - Università di Torino
andrea.valle@unito.it

Mauro Lanza
IRCAM-Paris
mauro.lanza@ircam.fr

ABSTRACT

Systema Naturae is a cycle of four compositions written for various instrumental ensembles and electromechanical setups. The workflow includes design and construction of electromechanical instruments, algorithmic composition, automated notation generation, real-time control of the setups and synchronization with the acoustic ensemble during the performances. These various aspects have to be integrated in a unique working pipeline, that has to be shared between the two authors, and thus requires to define communication protocols for sharing data and procedures. The paper reports on those aspects and on the integration required between hardware and software, non-real time and real time operations, acoustic and mechanical instruments, and, last but not least, between the two composers.

1. INTRODUCTION

The cycle *Systema Naturae* (2013-17, [1]) includes four 20 minute pieces. Its main feature is the use of acoustic instruments together with computer-controlled physical objects placed in a specific spatial organisation [1] .

Two references are at the base of whole cycle. The first is the Medieval tradition of bestiaria, herbaria and lapidaria intended as multi-faceted catalogues of miscellaneous beings. A second reference for the work is taxonomy, that is, the systematic description of living organisms that dates back to Linnaeus' *Systema Naturae* (hence the name of the whole cycle) as the rationalistic possibility of ordering the polymorphous (and many times teratomorphic) appearance of nature. Each piece of the cycle is organised as a catalog of short pieces that receive an invented Latin name in binomial nomenclature (see two examples in Figure 11 and 14). *Regnum animale* (RA), featuring a string trio and 25 objects, has been commissioned by Milano Musica Festival and premiered by RepertorioZero (2013). *Regnum vegetabile* (RV), for sextet, includes 30 hacked hair dryers, and has been commissioned by Ensemble Mosaik (Darmstadt, 2014). *Regnum Lapideum* (RL), for septet and 25 objects is a commission by Ensemble 2e2m and has been premiered in Paris (2016). The cycle is intended to have a final chapter, *Fossilia*, to be premiered in 2017. *Fossilia* is a conclusive piece, to be played together with all the *Regna* as the final chapter of the *Systema*, and it is scored for all the players and the three setups. Figures 1,2,3 show respectively RepertorioZero premiering RA, RV setup installed during a performance by ensemble mosaik, Ensemble 2e2m rehearsing RL.

In the following, we will not discuss algorithmic composition techniques, rather we will focus on instrument design and construction, and on shared communication protocols for music composition in a physical computing environment.

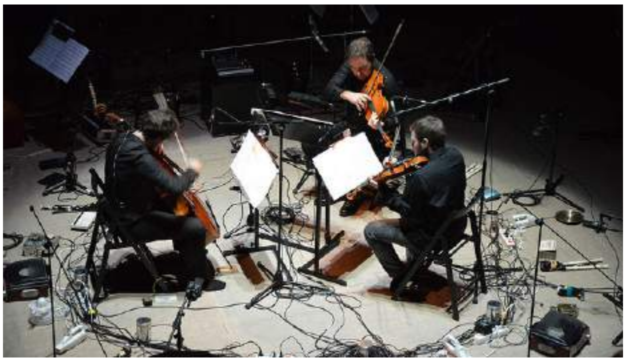


Figure 1. RepertorioZero premiering *Regnum animale*.

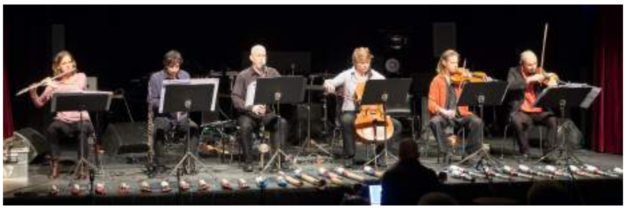


Figure 2. ensemble mosaik performing *Regnum vegetabile*.

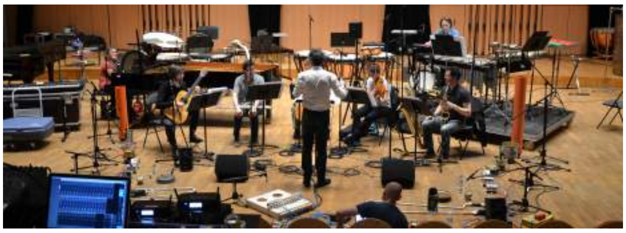


Figure 3. Ensemble 2e2m rehearsing *Regnum lapideum*.

[1] Photo/video documentation can be found at http://vimeo.com/vanderaalle and http://www.flickr.com/photos/vanderaalle/albums

2. ELECTROMECHANICAL INSTRUMENTARIUM

The main idea at the basis of the electromechanical setups is to create “residual orchestras” [2], that is, ensemble of instruments made up of debris and re-cycled objects (hence on, sound bodies). Residual orchestras belong to two worlds, as they are both everyday objects with a mechanized control and offer an acoustic behaviour similar to music instruments. The main ratio at the basis of the composition of *Systema naturae* is thus to create and explore a middle ground where mechanized objects can be controlled in a standard –even if basic– musical way (by creating events, exploring their spectra, organizing their dynamics) while music instruments are treated in an “object-like” fashion by means of a wide usage of extended techniques. As an example, string trio and guitar include strings prepared with Patafix glue pads to create inharmonic spectra, while wind instruments largely use multiphonics and percussive sounds. To be successful, sound integration between objects and instrument has been based on a complete analysis of sounds, recorded from both objects and instruments, so that acoustic information could be stored and used as a sort of homogenous ground while composing. Audio simulation of the pieces has proven fundamental in avoiding misconception about acoustic behaviour and in interactively exploring an unusual acoustic territory (see Section 3). Given such a technological infrastructure, it has then become possible to exploit a wide range of “classic” algorithmic techniques for composition (e.g. cellular automata, canonic techniques, data sonification, etc.). On the same side, a classification of objects in ethnomusicological terms (see in the following) has proven useful in creating a common conceptual ground with acoustic instruments.

Residual orchestras are meant to challenge the notion of instrument by reducing it to its minimal root. The seminal classification by Hornbostel and Sachs (H-S) [3] has prompted a vast, still on-going, debate among ethnomusicologists about the nature of music instrument [4]. By taking into account also electronic and digital music instruments [5] [6] it is possible to propose a minimal definition, that considers a music instrument as a device capable of generating sound once a certain amount of energy is provided. Three elements are thus relevant: the physical body, an energy source, and a control interface that allows a (variably) fine tuning of the latter, so that the physical body can respond properly. In residual orchestras, physical bodies are designed and assembled following three main principles, inspired by sustainable design ([7]; [8]; in particular here [9]): refabrication, softening, flexibility.

Refabrication: many practices around the world have traditionally developed specific attitudes towards the “refabrication” of objects as a normal way of shaping and reshaping the semiotic status of material culture [10]. A first example of residual orchestra by the author is the Rumentarium project [11].

Softening refers to the sound body hardware: being so simple and intrinsically costless, sound bodies can be “produced while designed” in an improvisation-like mood, starting from available materials. As a consequence, their hardware nature is quite “soft”: sound bodies, and their parts, can be replaced easily and effortlessly. Sound bodies in most cases remain open, that is, accessible for manipulation. All the orchestras typically present a no-case look, overtly showing components and connections.

Flexibility is here intended as the capability of the residual orchestras to be modified in relation to specific needs: as an example, a performance may require a certain setup in relation e.g. to the presence of microphones for the amplification of sound bodies. Orchestras are assembly of various objects, thus they are made of modular components, that e.g. can be easily replaced.

While the previous considerations may apply to a variety of sound instruments based on a DIY approach, both acoustic [12] and electronic [13], in residual orchestras the energy source does not involve at all the human body (an important feature in ethnomusicological classification, [4]), as it is electromechanical, ad typically (even if not always) exploits low voltage motors/actuators. This feature is crucial in bridging sound bodies with computational control, that covers the third element of the previous minimal definition of music instrument. Since 2000 microcontroller boards (e.g. Arduino) have played a pivotal role in physical computing [14], not only in providing an interface between software and the physical environment, but have prompted a new design perspective [15], that has revitalised the DIY technological community. At the moment, many options are indeed available, including the flourishing of single-board computers, like Raspberry PI, UDOO or Bela. In the context of residual orchestras, physical computing provides the control layer for instrument behaviour. Physical computing also implements the principle of flexibility in relation to information processing and symbolic manipulation, as a computer’s main strength is that it ensures not only programming but also re-programming. In short, by means of computational control, residual orchestras allow to create an “acoustic computer music” [11]. *Systema naturae* is entirely based on various residual orchestras that provides an apt correlate of the main aesthetic assumption at the base of the cycle, nature as a catalog of heterogeneous –and most time bizarre– entities, that, yet, can be used in relation to advanced algorithmic composition techniques. The instrumentarium prepared for *Systema Naturae* has to respect three complex constraints:

Low cost construction and maintenance: technical budget is typically very limited and must be distributed (by design) over a large set of sound bodies, the building of each of them having thus to cope with a severely reduced average budget. Typically, budget does not allow to outsource instrument building: setups are thus entirely designed and built by the authors. A DIY approach is fundamental as, due to the nature of sound bodies, maintenance is required while installing setups for concerts.

Transportability: for each piece, the whole setup has to be “designed for disassembly” [9], as it must be assembled/disassembled easily and quickly as possible in order to perform the piece in various locations and in the context of concerts including other complex setups. All technical materials (sound bodies, physical computing interface, powering units, cables) should be included, to avoid issues (and additional cost) on location with rented materials. Due to the hacked nature of the materials, some spare parts should be included in the transported package. Moreover, the overall volume and weight of the technical material must be reduced to be easily transportable. Since their premieres, both RA and RV have been performed more than 10 times in various European locations.

Time responsivity: the previous two constraints necessarily reduce the complexity of sound body, so that their behaviour is extremely simple, in many cases resulting in the production of a single fixed sound with unvarying spectral and dynamic features. To cope with this from a composition perspective, it is crucial to be able at least to finely tune their temporal behaviour. This means that sound bodies must provide a fast attack and decay, so that complex rhythmical organization becomes possible.

In the following, we describe various designs for sound bodies. As noted by [4], the classification of music instruments can be quite complex if only considering sound production, while, on the other side, it can be extended to include various criteria depending on specific ethnographical practices. The classic taxonomic organisation by Hornbostel and Sachs [3] (H-S) can be tailored to fit generic needs by distinguishing among macrofamilies of instruments on the base of mechanical features. In fact, other criteria may be applied resulting in a multidimensional classification (as proposed by [5]). In our case:

Control behaviour: a basic distinction opposes sound bodies with discrete behaviour (on/off, D) to those allowing a continuous one (C).

Control technology: most sound bodies are operated via microcontroller (M), but a subset requires a soundcard (S). While this is not relevant from the user perspective, thanks to software abstraction, nevertheless this implies a specific different technical implementation. In turn, microcontrollers can be used to activate 12 V DC/AC devices or $>$ 12 V (mostly, 230 V) AC ones, a feature that requires specific attention.

Pitch: an important feature in order to compose with sound bodies is the opposition between pitched and unpitched ones (P/U), as pitched sound bodies may be tightly integrated with harmonic content provided by acoustic instruments.

Setup context: sound bodies have been designed in relation to each piece of the cycle and its instrumental setup. RA includes a string trio, surrounded in circle by an amass of computer-driven, electro-mechanical devices built from discarded and scavenged everyday objects and appliances (electric knives, radio clocks, turntables, and so on). RV' setup includes a traditional acoustic instrument sextet (string and wind trio), placed behind a 7-meter line of 30 wind sound bodies (vs. RA's circle), all using the same model of hair dryer. In RL the instrumental ensemble is focused on percussive and plucked instruments (two strings, two woodwinds, guitar, piano and percussion), and the same happens for the electromechanical setup that includes plucked strings and percussions, scattered on the floor. While RA was mostly heterogenous and based on house appliances, and RV focused on wind instruments, RL favours a metallic, percussive approach.

In the following, sound bodies are described following H-S basic principles (see in particular [16]), adding specific tags from the previous classification schema. The subscripts in the Control and Setup tags indicate respectively the current voltage and type (only if V = 12 and type = AC) and the number of occurrences in the specified piece. Figure 4 shows the resulting multidimensional classification, in which only the parts of the H-S tree relevant for sound bodies are represented (e.g. membranophones are not represented). Sound bodies have been named following a sort of distortion process applied to various references to instrument/object name.

I. Idiophones

The idiophone family consists of objects that directly produce sounds by their whole body.

Cono (D, S, U, RL_8): "coni" are 8 woofer loudspeakers to be placed directly on the ground. Each cono is surmounted by an object, simply placed on its top. Loudspeakers are intended to deliver audio impulses that make the surmounting object shake or resonate. Objects include various metal boxes and two large water PVC pipes, having the diameter of the woofer and respectively 80 and 112 cm long. They act as resonators, filtering the impulses provided, with a clear kick drum sound. (see the two orange pipes in Figure 3).

Lampadina (D, M_{230}, U, RA_1): a light bulb that can be turned on/off via a relay. It allows to exclusively hear the sonorous relay click while providing at the same time a visual rhythmic cue.

Meshugghello (D, $\mathrm{M}_{12\mathrm{AC}}$, P, RA_1) an AC doorbell, that provides a metallic, intermittent pitched sound.

Cimbalo (C, M_{12}, U, RL_2): "cimbali" are scraped instruments in which a plastic beater variably rotates over a metal plate (e.g. a pan).

Sistro (C, M_{12}, U, RL_2): a vessel rattle in which the beater rotates inside a metal box, causing the scraping of various elements (e.g. buttons, seeds, rice) on the internal surface.

Molatore (C, M_{12}, U, RA_4): literally "grinder", it is a scavenged component from tape decks (typically, cassette or VHS players) that generates low-volume, noisy, continuous sounds by rubbing against various metallic surfaces.

Spremoagrume (D, M_{230}, U, RA_1): a juicer featuring a low-pitched rumble caused by friction between its rotating element and the container.

Rasoio (D, M_{12}, U, RA_1): an electric shaver put into a metal box partially filled with buttons, that provides a buzzing, almost snare-like sound (Figure 5-4).

Segopiatto (D, M_{230}, U, RA_4): a "segopiatto" is built by shortly exciting a cymbal (or, in general, a metal resonating plate) by means of an electric knife. The knife's blade is leaning above the cymbal. The knife's motor is operated at 230 V and switched on/off by a 12 V relay. Figure 5-3 shows a segopiatto on a cymbal.

Tola (D, M_{12}, P, RL_6): an idiophone made up of a metal bar screwed into a metal can (Figure 7). Once the bar is struck, the metal can acts as a resonator, the resulting sound including both spectral components from the bar and the

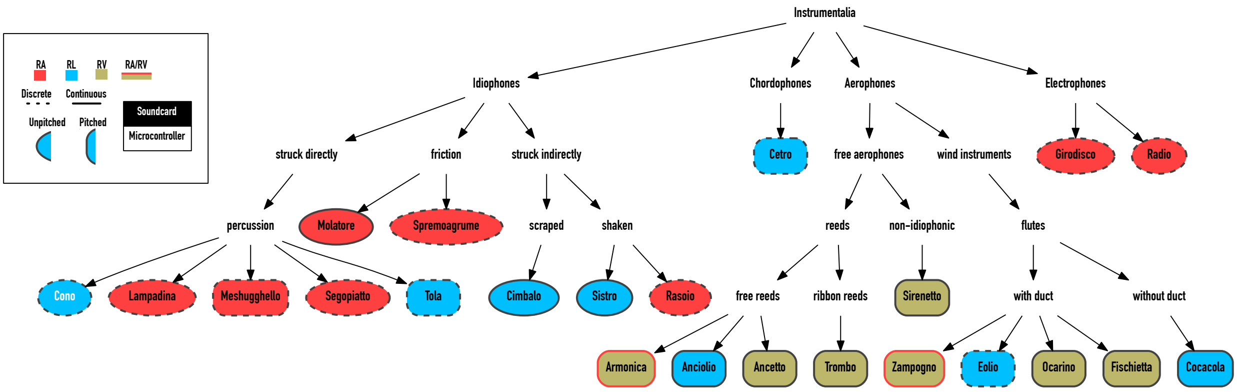


Figure 4. Taxomomy of sound bodies.

can. In order to struck the bar, an eccentric (Lego) wheel has been connected to the motor axle, thus hitting the bar once per rotation. To solve the complex stand issue [12], each tola is loosely glued to the floor, and the motor is supported by small metal bars again taped on the floor.

II. Aerophones

Aerophone construction is particularly affected by design constraints. In relation to air pressure for wind instruments, compressors have been widely used to in automated instruments (e.g. by artists like Marcel·lí Antúnez Roca, Cabo San Roque, Jean-François Laporte) but could not be taken into account because of their dimension and weight. As a general (and low cost solution), modified hair dryers have been used for air generation with controllable, variable pressure. The heating resistor is removed from hair dryers, that can thus be directly powered by a 12 V current. As a result, they are typically not capable of delivering the pressure required to activate reeds, with some exceptions.

Trombo (C, M_{12}, P, RV_{10}): as low pressure is an issue to be taken into account, trombi have been designed starting from football toy trumpet, that features a very thin, and thus efficient, plastic membrane. In this sense, they are reed instruments. Trumpet horns have been modified in various ways, e.g. extended by means of water PVC pipes, including curves for easier layout, reaching a low B (≈ 61.7 Hz). Figure 6, *a* to *d* shows respectively the toy trumpet mouthpiece glued to the hair dryer, the horn, a PVC pipe extension, the use of curves.

Armonica (C, M_{12}, P, RA_3-RV_6): they are built by connecting a harmonica to a hair dryer (Figure 5-5). As the direction of the flow of air depends on motor polarity, by inverting the wiring on the motor it is possible to use both blow and draw reeds.

Anciolio (C, M_{12}, P, RL_1): a single, large accordion reed (C♯, ≈ 69.3 Hz) is accommodated inside a metal can, on top of which a computer fan pushes the air flow.

Ancetto (C, M_{12}, P, RV_1): a small toy reed instrument (Figure 6, *g*).

Zampogno (C, M_{12}, P, RA_4-RV_6): a "zampogno" is a set of three recorders connected to a modified hair dryer, acting like a bagpipe (Figure 5-1). A zampogno may include various recorder sizes, from tenor to sopranino. Some standard tunings produce microtonal alteration due to hair dryers' air pressure.

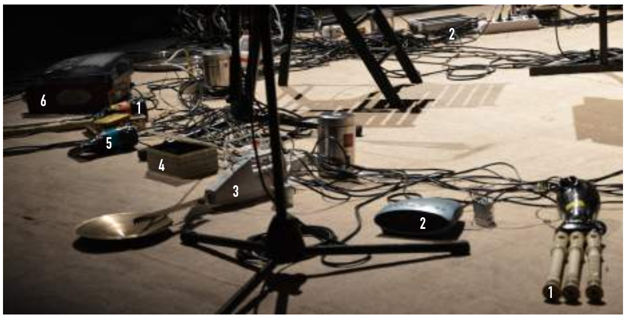


Figure 5. Sound bodies in *Regnum animale*.

Eolio (D, M_{24}, P, RL_4): a hair dryer is used as source for a PVC pipe flute, but, in order to increase air pressure, it is operated at 24 V, and switched on/off via relays. The resulting wind instrument produces high pitched, gliding aeolian sounds, its actual pitch depending both on pipe length and on the duration of the provided air packet.

Cocacola (C, M_{12}, P, RL_1): a bottle of Coca-Cola blown directly on its edge, with a clear pitch.

Sirenetto (C, M_{12}, U, RV_2): three toy jet whistles packed so to be operated via a single hair dryer (Figure 6, *f*).

Ocarino (C, M_{12}, P, RV_2): two plastic ocarinas packed so to be blown by a single hair dryer (Figure 6, *h*).

Fischietta (C, M_{12}, P, RV_1): a toy whistle blown by hair dryer (Figure 6, *e*).

III. Chordophones

The only member of this group is the "cetro", but it is also the most complex sound body used in the cycle. The **Cetro** (D, M_{12}, P, RL_1) is an automated zither. Inspired by instruments like the cymbalom, it features a wooden box with 12 electric bass steel strings. The cetro is tuned by tones from low C♯ (≈ 69.3 Hz) to A♯ (220 Hz), with a semitone between the sixth and seventh string, thus distributing twelve chromatic pitches over two octaves, in order to wide the available register. In order to pluck the strings, twelve 12 V DC motors, one for each string, are suspended over a bridge. As strings present a large gauge, reduced gear motors are in use, to ensure proper torque. Due to the inelastic response of the axle, flexible rubber picks have been preferred to stiffer (e.g. vinyl) one, to avoid breakage.

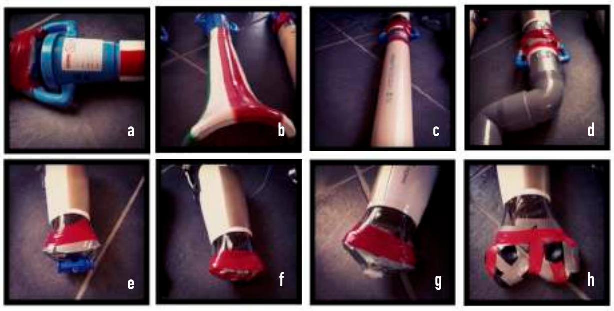


Figure 6. Some wind instruments for *Regnum vegetabile*.

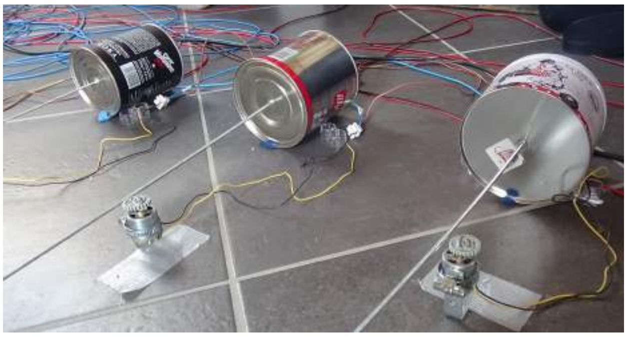


Figure 7. “Tole” (It. plur. for “tola”).

IV. Electrophones

The electrophone extension proposed by Galpin to the original H-S taxa is notoriously problematic and the following examples are not exceptions. In this sense, it has been suggested that the electronic generation can be intended as a modification of existing categories [17]. In our case, on one side all sound bodies are electromechanical in terms of energy supply, on the other side, sound bodies that are strictly based on electronic sound production, like the following ones, neither are inserted into an electroacoustic control chain (they generate sound through internal amplification, as autonomous objects) nor are controlled in a specific way.

Girodisco (D, M_{230}, U, RA_3): low quality, self-amplified turntables with a direct connection between the motor and the rotating plate (Figure 5, 6). The relay opens/closes the motor powering. This results in a gliding behaviour, both in speeding up and down the plate. In order to emphasise the gliding sound, each girodisco plays a 33rpm disc at 45rpm. Content of the disc is irrelevant, as what can be heard is the gliding tone, almost uncoupled from the played vinyl.

Radio (D, M_{230}, U, RA_3): a small radio clock in which the loudspeaker is interrupted by an associated relay (Figure 5, 2). By open/closing the audio signal going to the loudspeaker, a clicking burst of sound is obtained. In the case of radios, a very fast attack is thus possible. Radios are tuned on a frequency where no signal is detected (white or hum noise).

From the previous behaviour description, it is clear that girodisco and radio could be considered respectively as friction and struck idiophones.

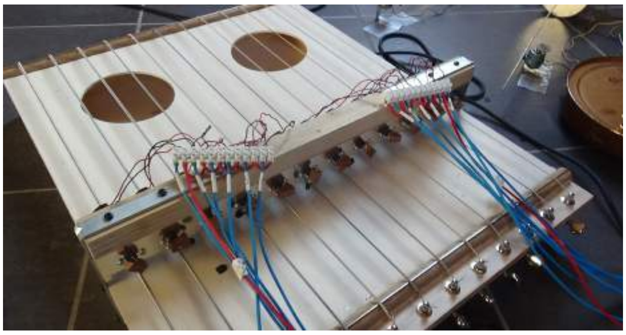


Figure 8. The cetro.

3. ALGORITHMIC COMPOSITION PRACTICES AND SHARED PROTOCOLS

The design of the sound bodies (and of their control procedures) is a crucial part of the work, but it has a counterpart on composition. Music composition, at least if referring to the Western corpus of classic practices related to acoustic instruments and Common Practice Notation (CPN), is typically a one-man activity, the composer working by her/himself. On the contrary, *Systema Naturae* is a shared effort with a minimum division of labour between the two authors. Composition work includes the design and construction of electromechanical instruments (see before), algorithmic composition, automated notation generation, real-time control of sound bodies and their synchronization with the acoustic ensemble during the performances. These various aspects had to be integrated in a single working pipeline, thus requiring to define communication protocols for sharing data and procedures in relation to all the aforementioned aspects. Moreover, the authors live in different countries, and large part of the work had to be realised remotely via internet (e.g. via GitHub, file sharing services, VoIP): in the case of RL the authors never met face to face while composing. This working situation was particularly complex as it included experimentation with physical objects (i.e. the sound bodies) and their behaviour. Finally, while sharing a common attitude towards algorithmic composition, the authors favour different programming paradigms and languages (respectively, functional programming in Lisp via Open Music –OM– for ML [18], and object-oriented programming in SuperCollider –SC– for AV [19]).

The composition work can be described as a three step process: architectural design, sound organisation, performance design. In Figure 9 the three steps are represented vertically (bold rectangles), activities by the two composer are placed on opposite sides (ML and AV), while the dotted area in the middle represents contents that are shared over internet (shared information), crossbreeding both sides. Rectangles inside step 2 represent, more than objects, activity modules that result in data outputs. Grey rectangles indicate outputs that are passed to step 3. In the following discussion, numbers refer to Figure 9.

1. Architectural design

A pre-production phase that focuses on the meta-organisation of a piece, including general organisation of the composition (e.g the catalog form, the average duration of each

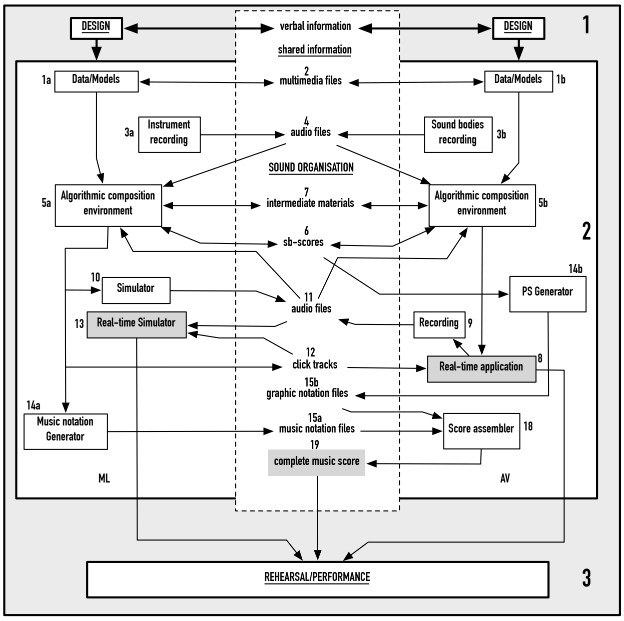


Figure 9. Shared composition information flow.

piece, the choice of acoustic instruments and timbre), the design and building of sound bodies, the development of software to be used while composing (e.g. music notation generation, sound analysis, sound resysnthesis for simulating the result).

2. Sound organisation

The core step concerns indeed the organisation of sound materials, that typically includes various algorithmic techniques (see Data/Models 1a-b). Shared output includes data files, algorithms and implementations, graphics etc (2). In order to clearly foresee composition results, sound bodies are recorded (3b) and sound files shared (4). The same happens for instrumental samples (3a), that are mostly recorded from scratch (rather than imported form libraries), as they include special playing techniques. These sound files (4) can be used to feed various algorithms while composing, e.g. audio analysis to gather spectral data (in the Algorithmic composition environments, 5a-b, respectively in OM –Figure 10– and SC). The main output of the two Algorithmic composition environments are "sb-scores" (i.e. scores for sound bodies, 6). Sb-scores are ASCII files that specify events to be triggered in the sound bodies. Sb-scores are defined by the following protocol, loosely inspired by Csound score file format [20]:

```
ocarino1  79.876 0.036 210 210
zampogno7 37.556 0.074  50  50
```

where each line specifies an event by means of instrument name, start time and duration (in seconds), initial and final values (as integers with a 8-bit resolution). Sb-scores represent the main interchange format as they can be parsed on both side in order to be modified (e.g. for post-processing, or to be analysed in order to include acoustic instruments), thus they are fed back into each Algorithmic composition environment (5a-b). Similar scores, even if incomplete, are exchanged as intermediate materials (7), e.g. as composition sketches including numerical specifications (like in sb-scores) for events to be performed by acoustic instruments. Intermediate materials include indeed a very heterogenous collection of working materials. Sb-scores are the final output of composition for sound bodies, but they have also to be processed (5b) to obtain the final data format for real-time usage (see later), and then fed into the Real-time Application (8). Sb-scores are played on the electromechanical setups and recorded (9), thus providing feedback on final composition results for sound bodies. A Csound-based, non real time Simulator (10, see [21]) is able to include all the sound samples (both of sound bodies and acoustic instruments) to generate an audio file with a complete and accurate simulation of the piece (11). The Algorithmic composition environment (5a) generates automatically click tracks via OM (12, i.e. audio files). Click tracks are used to synchronise musicians with sound bodies and have to be played back during live performance: they are thus integrated into the Real-time application (8). Another component, the Real-time Simulator (13) performs a different function, as it plays back sound files of the sound body parts of pieces, syncing them with the relative click tracks on a separate channel. Its main usage is during rehearsals with musicians, and proved fundamental, as setups cannot be easily transported, and simulation of sound bodies' parts provides accurate feedback for performers while rehearsing. The Music notation generator (14a, via OM [22]) is used to generate CPN scores for instruments (music notation files, 15a), while the PS Generator (14b) creates automatic PostScript visualization from sb-score parsing (graphic notation files, 15b), that gives a visual overview of the sound body parts (Figure 11). In general, automatic music notation generation is a complex task but it is instrumental in ensuring an integrated algorithmic composition (IAC) pipeline, rather than a computer assisted one (CAC) [23]. These two graphical outputs are then assembled automatically by the Score assembler (18), that generates ConTeXt files [24], a TEX-based typesetting macropackage preferred to LATEX for its flexibility [23]. The resulting final complete music score (in PDF, 19) can then be delivered to musicians.

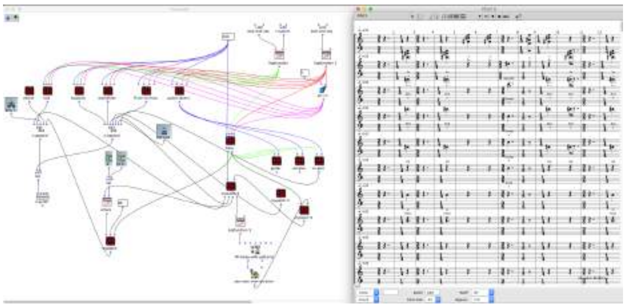


Figure 10. Algorithmic composition environment (OM).

3. Performance design

A specific post-production step is dedicated to design and development (hardware and software) of the real-time performance, particularly delicate as it includes musicians and sound bodies. Rehearsing and live situations require to develop control strategies as robust and flexible as possible.

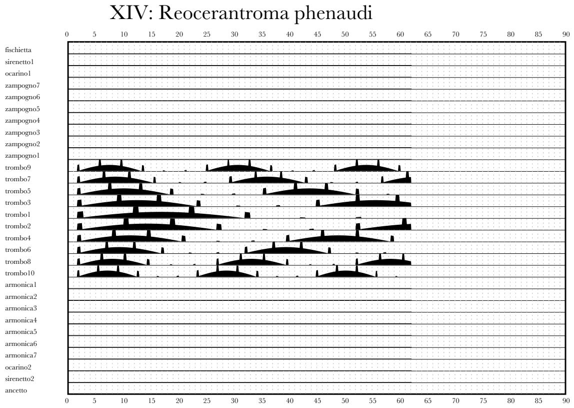


Figure 11. PostScript automatic sound body notation.

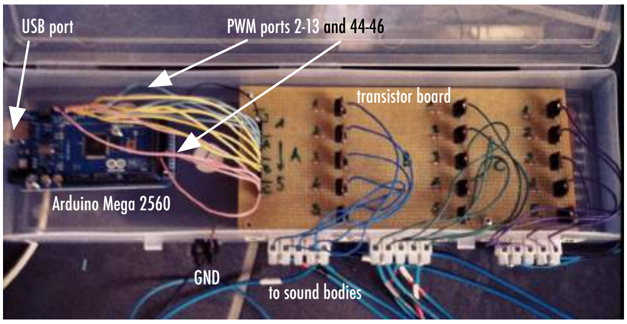


Figure 12. Arduino and transistor board.

This step includes GUI development for sound body control and simulation with click tracks.

In *Sistema Naturae* all the control is demanded to the main computer, as many high-level processes are required, including parsing of textual scores, real-time audio generation for click tracks, GUI creation for live performing. The pieces are all performed live, but there is not interaction between sound bodies and musicians. The main DAC components are Arduino boards converting digital control data (via PWM in case of "continuous" control) in analog signals that drive a circuit based on a Darlington TIP120 transistor (or similar, see [14]). The transistor board then modulates current to feed the electromechanical components of the objects. The operating voltage is 12 V, that allows to directly drive small DC motors or to trigger relays that can be used also to switch on/off 230 V AC devices. Figure 12 shows an Arduino Mega 2560 encased with its transistor board. In this case (RV), only the 15 PWM ports are used (as indicated). In case of loudspeakers, a sound card acts as a DAC, generating audio signal (in form of impulses), then delivered to an amplifier and finally to loudspeakers. Figure 13 depicts the flow of information for the real-time performance of the pieces, in the general case of RL that includes also audio generation for coni (loudspeakers). The software (in grey) is written in SuperCollider, as the latter provides a high-level language, GUI generation, audio synthesis and playback [19]. It features two components, the Controller and the Scheduler (in solid lines). In Figure 13, GUI elements are dash-dotted, data are dashed, normal lines represent control information, bold lines audio information, solid line rectangles are hardware components (already discussed). In the SuperCollider application, the Controller component accesses the Arduino boards, opening the USB ports, and in the case of audio generation, it is also interfaced to audio synthesis for impulse generation (synths are standard SuperCollider objects for sound synthesis). A GUI element allows easy control over each instrument's behaviour for checking, tuning etc. The Scheduler is responsible for real-time performance of the pieces. It delivers time-stamped information to the Controller, that, in turn, activates the sound bodies. In order to be sequenced in real-time, sb-scores are post-processed and converted into a new format by means of a sampling process with a sample period $T = 0.01$ seconds. A note event, defined in the sb-score by its initial and final values over a duration, is linearly sampled. Sampling is required by the Controller as the latter can only deal with the setting of a specific value for a port in Arduino. Sb-score sampling results in a $sb \times (100 \times d)$ matrix, where sb is the number of sound bodies and d is the duration in seconds of each sb-score. During real-time scheduling, every 10 ms a bundle of n messages is sent to the Arduino boards. A control is performed so that, if the sound body is not playing or its value has not changed, the relative message is not sent. This means that the bundle has a maximum size of sb (all sound bodies playing) and a minimum of 1 (only one instrument), while in case of silence no message is sent. Sampling solves an issue related to having many parallel process (i.e. "notes" in "voices") by allowing to implement a single scheduling process with a fixed computational cost (and a maximum network cost). The Scheduler loads the click audio files (one for each piece), the post-processed sb-scores, and connects itself to the Controller module. Its GUI allows complete control over the score scheduling and is provided as the main interface for the performer. Figure 14 shows the scheduler GUI for *Fossilia*. It allows to load a score by referring to its progressive number, and to play/stop the score. For sake of simplicity, in live performances –after a score has been played back– the next score is loaded automatically.

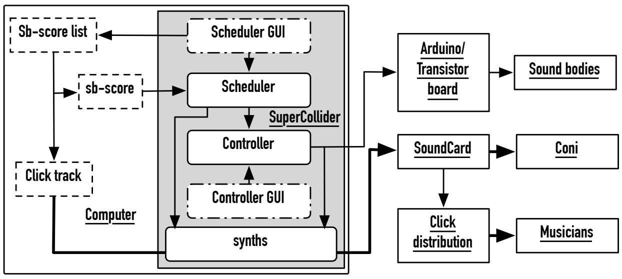


Figure 13. Real-time information flow.

4. CONCLUSIONS

Systema naturae is a four year effort that has required to integrate elements and activities that are typically kept separate: sound bodies had to meet various criteria, from design to packaging, from music control to live synchronisation

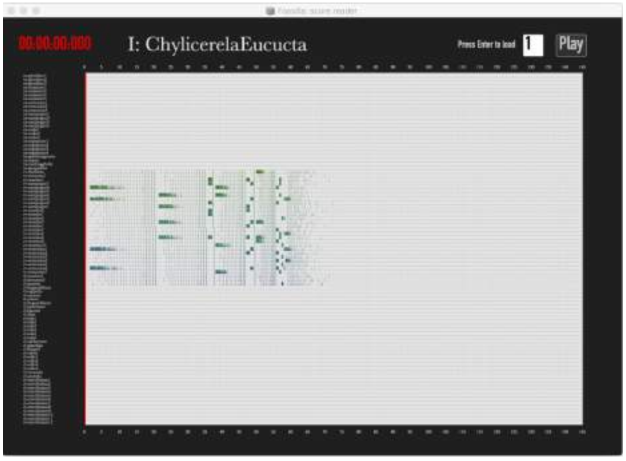


Figure 14. Scheduler GUI for *Fossilia*.

with musicians; all activities, ranging from low- (object construction) to high level (music composition) –passing through physical computing organisation– had to be shared between the authors. All these constraints have prompted to establish a clear working methodology that is mostly unusal in contemporary composition.

Acknowledgments

We thank RepertorioZero, ensemble mosaik and Ensemble 2e2m for the commissions, Finder for relais in RA, Liuteria Alvermann for the cetro, Simone Pappalardo for the anciolio and Francesco Richiardi for technical support.

5. REFERENCES

[1] P. Roullier, Ed., *Mauro Lanza et Andrea Valle: Systema naturae*. Champigny-sur-Marne: Ensemble 2e2m, 2016.

[2] A. Valle, "Residual orchestras: Notes on low profile, automated sound instruments," in *Proc. of the Cumulus Conf. '15*. Milano: McGraw-Hill, 3-7 June 2015, pp. 717–729.

[3] E. M. v. Hornbostel and C. Sachs, "Classification of musical instruments." *The Galpin Society Journal*, vol. 14, pp. 3–29, 1961.

[4] M. Kartomi, "The classification of musical instruments: Changing trends in research from the late nineteenth century, with special reference to the 1990s," *Ethnomusicology*, vol. 45, no. 2, pp. 283–314, 2001.

[5] R. T. A. Lysloff and J. Matson, "A new approach to the classification of sound-producing instruments," *Ethnomusicology*, vol. 29, no. 2, pp. 213–36, 1985.

[6] S. Jordà, "FMOL toward user-friendly, sophisticated new musical instruments," *Computer Music Journal*, vol. 26, pp. 23–39, 2002.

[7] W. McDonough and M. Braungart, *Cradle to Cradle : Remaking the Way We Make Things*. New York: North Point, 2002.

[8] J. F. McLennan, *The Philosophy of Sustainable Design*. Kansas City: Ecotone, 2004.

[9] P. Tamborrini, *Design sostenibile. Oggetti, sistemi e comportamenti*. Milan: Electa, 2009.

[10] S. Seriff, *Recylced Re-seen*. Santa Fe: Museum of International Folk Art, 1996, ch. Folk Art from the Global Scrap Head: The Place of Irony in the Politics of Poverty, pp. 8–29.

[11] A. Valle, "Making acoustic computer music: The rumentarium project," *Organised Sound*, vol. 18, no. 3, pp. 242–254, 12 2013.

[12] B. Hopkin, *Musical Instrument Design: Practical Information for Instrument Design*. Tucson: Sharp Press, 1996.

[13] N. Collins, *Handmade Electronic Music. The art of hardware hacking*. New York–London: Routledge, 2006.

[14] D. O'Sullivan and T. Igoe, *Physical Computing*. Boston, Mass.: Course Technology, 2004.

[15] M. Banzi, *Getting started with Arduino*. O'Reilly, 2009.

[16] "Revision of the Hornbostel-Sachs Classification of Musical Instruments," MIMO, http://www.mimo-international.com/documents/HornbostelTech. Rep., July 2011.

[17] M. B. Bakan, W. Bryant, G. Li, D. Martinelli, and K. Vaughn, "Demystifying and calssifying electronic music instruments," *Select Reports in Ethnomusicology*, vol. 8, pp. 37–64, 1990.

[18] J. B. C. Agon, G. Assayag, Ed., *The OM Composer's Book*. Paris: Editions Delatour France/IRCAM, 2006, vol. 1.

[19] S. Wilson, D. Cottle, and N. Collins, Eds., *The SuperCollider Book*. Cambridge, Mass.: The MIT Press, 2011.

[20] R. Boulanger, Ed., *The Csound Book*. Cambridge, Mass.: The MIT Press, 2000.

[21] M. Lanza, "La libreria OpenMusic om4Csound," in *Prossime distanze. Atti del XVIII CIM*, A. Valle and S. Bassanese, Eds., AIMI. Venezia: DADI, 2011.

[22] J. Bresson, C. Agon, and G. Assayag, "Openmusic: Visual programming environment for music composition, analysis and research," in *Proceedings of the 19th ACM MM*. New York, NY, USA: ACM, 2011, pp. 743–746.

[23] A. Valle, "Integrated algorithmic composition," in *NIME 208: Proceedings*, 2008, pp. 253–256.

[24] H. Hagen, *ConTeXt the manual*, PRAGMA Advanced Document Engineering, Hasselt NL, 2001.

GENERATING EQUIVALENT CHORD PROGRESSIONS TO ENRICH GUIDED IMPROVISATION : APPLICATION TO RHYTHM CHANGES

Ken Déguernel[1,2] **Jérôme Nika**[2] **Emmanuel Vincent**[1] **Gérard Assayag**[2]
[1]Inria, F-54600 Villers-lès-Nancy, France
[2]IRCAM STMS Lab (CNRS, UPMC, Sorbonne Universités)
ken.deguernel@inria.fr jerome.nika@ircam.fr
emmanuel.vincent@inria.fr gerard.assayag@ircam.fr

ABSTRACT

This paper presents a method taking into account the form of a tune upon several levels of organisation to guide music generation processes to match this structure. We first show how a phrase structure grammar can represent a hierarchical analysis of chord progressions and be used to create *multi-level progressions*. We then explain how to exploit this multi-level structure of a tune for music generation and how it enriches the possibilities of guided machine improvisation. We illustrate our method on a prominent jazz chord progression called 'rhythm changes'. After creating a phrase structure grammar for 'rhythm changes' with a professional musician, the terminals of this grammar are automatically learnt on a corpus. Then, we generate melodic improvisations guided by multi-level progressions created by the grammar. The results show the potential of our method to ensure the consistency of the improvisation regarding the global form of the tune, and how the knowledge of a corpus of chord progressions sharing the same hierarchical organisation can extend the possibilities of music generation.

1. INTRODUCTION

In jazz music, and more generally in improvised music, the inherent form of a chord progression is often composed upon several levels of organisation. For instance, a subsequence of chords can create a tonal or modal function and a sequence of functions can be organised as a section. This multi-scale information is used by musicians to create variations of famous chord progressions whilst keeping their original feel. In this article, we define as *equivalent* two chord progressions sharing the same hierarchical organisation. We call a *multi-level progression* an entity describing an analysis of a music progression upon several hierarchical levels (for instance chord progression, functional progression, sectional progression, etc.) that can be used to generate equivalent chord progressions. We want to design a musical grammar from which we can infer this hierarchical information and then use it to extend the possibilities of music generation.

Machine improvisation systems use style modelling to learn the musical style of a human improviser. Several methods of style modelling have been used for music generation including statistical sequence modelling [1–3], extended Markov models [4], methods using deep and recurrent neural networks [5, 6], and methods based on the factor oracle, a structure from formal language theory [7, 8] adapted to a musical context in [9]. Studies around style modelling have made this structure a proficient tool for improvised performance and interaction. These studies involve the creation of specific navigation heuristics [10], its adaptation to audio features [11, 12] or its integration in a system mixing background knowledge with the local context of an improvisation [13]. However, none of these systems takes the long-term structure of the improvisation into account.

Improvising on idiomatic music such as jazz [14], where the improvisation is guided by a chord progression, has been a major interest in machine improvisation. Donze et al. introduced the use of a control automaton to guide the improvisation [15]. Nika et al., with ImproteK [16], designed a system of *guided improvisation*, that is to say a system where the music generation model uses prior knowledge of a temporal scenario (e.g. a chord progression). The music generation model anticipates the future of a scenario while ensuring consistency with the past of the improvisation. It received positive feedback from professional improvisers and evolved according to musicians expertise [17]. However, these scenarios are purely sequences of symbols. The chord progression is only considered on a local level and it does not take into account the fact that the same harmonic progression can have different roles when played in different parts of a tune. Therefore, the improvisation does not adapt to the global structure. We would like the improvisation to take this global structure into account in a similar way as humans apply syntactic cognitive processes capable of handling the hierarchical structure of a song [18].

Form analysis, i.e. the problem of defining and analysing the global structure of a piece of music, is a major topic in Music Information Retrieval and Computational Musicology and it has been studied with methods from different fields. Giraud et al. use methods from bioinformatics and dynamic programming, based on the Mongeau-Sankoff algorithm [19], to perform fugue analysis [20] or theme and variations recognition [21]. Geometric approaches have also been used [22]; for instance, algorithms using the spiral array can be used to determine key boundaries [23] and therefore reveal tonal functions. Another approach using grammars was introduced in [24] for structure, rhythm and harmony analysis in tonal music. Some generative grammars can infer a hierarchical multi-level phrase structure [25]. This type of structure has been used in [26] to compute harmonic similarity of chord sequences. However, the above form analysis methods were used for music analysis only. Focusing on music generation instead, Steedman proposed a generative grammar based on rewrite rules for jazz music [27]. This grammar was used in ImproteK by Chemillier et al. [28] to create new instances of a given chord progression. However, we would like to use generative grammars to create a hierarchical analysis of a chord progression, thus creating a multi-level progression, and then directly involve it in the generative aspect of the improvisation.

In this article, we present two contributions. First, we propose a method formalising multi-level progression with phrase structure grammars. The structural aspect of equivalent chord progressions is analysed with a musician but the actual contents of the multi-level progression are then automatically learnt on a corpus. The grammar enables the system to generate multiple instances of a given progression rather than a single one and to provide its hierarchical structure. Second, we introduce a method using the information from a multi-level progression to expand the current methods of music generation. This method is based on the algorithms from [16] for music generation that we adapt to take into account the information from a multi-level progression; thus, creating a system able to improvise on a progression with changing voicing and considering its position in the structure of the tune. We apply this method to ‘rhythm changes’, the most played chord progression of jazz music, after the *blues* [29]. We first create a phrase structure grammar for ‘rhythm changes’, and then generate melodic improvisations over multi-level progressions provided by the grammar.

In section 2, we explain how a phrase structure grammar can model a multi-level progression by creating a hierarchical structure and we create a grammar for ‘rhythm changes’. Then, in section 3, we introduce a heuristic to generate a music sequence following a multi-level progression. In section 4, we describe our experiments and the results obtained with this new generation model. Finally, we conclude and propose some perspectives for this generation model in section 5.

2. GENERATIVE GRAMMAR AND SYNTACTIC STRUCTURE

In this section, we present how we can use a phrase structure grammar to create a multi-level progression from equivalent chord progressions. First, we present the definition of a phrase structure grammar, and then show an application to a jazz chord progression: the ‘rhythm changes’.

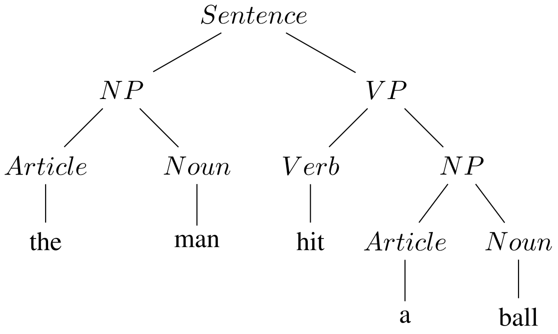


Figure 1. Diagram of the derivation of the sentence “the man hit a ball”.

2.1 Phrase structure grammar

A grammar $G = (N, \Sigma, R, s)$ is defined by:

- two disjoint finite sets of symbols: N the set of non-terminal symbols, and Σ the set of terminal symbols,
- a singular element $s \in N$ called the axiom,
- a finite set R of rewrite rules included in $(N \cup \Sigma)^* N (N \cup \Sigma)^* \rightarrow (N \cup \Sigma)^*$

where * is the Kleene star, i.e., X^* is the set of finite sequences of elements of X.

Phrase structure grammars are a type of grammar presented by Noam Chomsky, based on constituent analysis, that is to say on a breakdown of linguistic functions within a hierarchical structure. In [25], Chomsky presented this example of phrase structure grammar:

(i)	$Sentence \rightarrow NP + VP$
(ii)	$NP \rightarrow Article + Noun$
(iii)	$VP \rightarrow Verb + NP$
(iv)	$Article \rightarrow$ a, the...
(v)	$Noun \rightarrow$ man, ball...
(vi)	$Verb \rightarrow$ hit, took...

Rule (i) should be read as “rewrite $Sentence$ as $NP + VP$”, i.e., a *sentence* consists of a *noun phrase* followed by a *verb phrase*. Rule (ii) should be read as “rewrite NP as $Article + Noun$”, i.e., a *noun phrase* consists of an *article* followed by a *noun*, etc.

A *derivation* is the sequence of rules applied to create a specific sentence. Figure 1 shows a diagram representing the derivation of the sentence “the man hit a ball”. This diagram does not convey all the information about the derivation since we cannot see the order in which the rules were applied. Nevertheless, it clearly shows the hierarchical syntactic structure of this sentence, thus creating a visualisation of the constituent analysis.

$\boxed{A}$	I VI-	II- V^7	I VI-	II- V^7
	I I^7	IV IV-	I VI-	II- V^7
$\boxed{A}$	I VI-	II- V^7	I VI-	II- V^7
	I I^7	IV IV-	I V^7	I
$\boxed{B}$	III^7	III^7	VI^7	VI^7
	II^7	II^7	V^7	V^7
$\boxed{A}$	I VI-	II- V^7	I VI-	II- V^7
	I I^7	IV IV-	I V^7	I

Figure 2. Original chord progression of 'I Got Rhythm'.

2.2 Application to 'rhythm changes'

In order to test the formalisation of multi-level progressions with phrase structure grammars, we create such a grammar for a specific jazz chord progression: the 'rhythm changes'. We show how we can obtain a hierarchical analysis of the chord progression and that it can be used to generate equivalent chord progressions.

The 'rhythm changes' is a 32-bar chord progression used in George Gershwin's tune 'I Got Rhythm' ('rhythm changes' is short for "chord changes of 'I Got Rhythm'"). Figure 2 shows the original version of this chord progression. The main feature of this chord progression is its $AABA$ structure with a B section (the bridge) that contrasts sharply with the A section.

- The A section is an 8-bar structure with fast changing chords based on:
  - a series of *turnarounds*: a 2-bar function (for instance, I VI- II- V^7) affirming the tonic of the tune on bars 1&2, 3&4 and 7&8. We note τ a *turnaround* on the tonic. We also note τ_I as a restriction of *turnarounds* starting with a Ist degree chord ($\tau_1 \subset \tau$).
  - a short modulation to the IVth degree on bars 5&6. We note σ this modulation to the IVth degree.
- The B section is an 8-bar structure consisting of dominant seventh chords following the circle of fifths (III^7 VI^7 II^7 V^7). Each chord is played on a two bar span, giving a sense of key shifting. The improvisers usually emphasize this contrast, insisting on the guide notes (the 3rd and the 7th) of these dominant seventh chords. We note δ_i these 2-bar dominant seventh chords on the i^{th} degree functions.

The 'rhythm changes' is an interesting case study for our method because of the amount of variations existing around this chord progression. This is actually one of the main interests for musicians; the changes can be modified on the fly without discussion as long as the functions are present. They mix various versions of 'rhythm changes' as they improvise [29], the chord progression can be different at every iteration. Using a phrase structure grammar to generate 'rhythm changes' seems therefore appropriate. Considering chords, functions and sections as constituents, we can create a phrase structure grammar embedding the hierarchical structure of the 'rhythm changes' where chords are the terminal symbols.

In order to create this phrase structure grammar for 'rhythm changes', we analysed with a professional jazz musician all the 'rhythm changes' from the Omnibook corpus [13, 30]. This sub-corpus consists of 26 derivations of 'rhythm changes' and contains the melodies and annotated jazz chord symbols [31]. We present here the grammar that we created. This grammar has then been validated with jazz musicians (cf. section 4.1).

$$\begin{array}{rl}
(i) & Rhythm\ Changes \rightarrow A_1 + A_2 + B + A \\
(ii) & A_1 \rightarrow \tau_I + \tau + \sigma + \tau \\
(iii) & A_2 \rightarrow \tau_I + \tau + \sigma + \omega \\
(iv) & A \rightarrow A_1, A_2 \\
(v) & B \rightarrow \delta_{III} + \delta_{VI} + \delta_{II} + \delta_V
\end{array}$$

$\tau_I, \tau, \sigma, \omega, \delta_{III}, \delta_{VI}, \delta_{II}, \delta_V$ are learnt on the corpus

- Rule (i) shows the $AABA$ structure of the 'rhythm changes'. These 8-bar sections are the biggest constituents after the whole chord progression itself.
- Rules (ii) and (iii) show the composition of an A section in four 2-bar functions. An A section starts with a τ_I in order to highlight the beginning of the section with a Ist degree chord. Then, another τ is played, followed by the modulation to the IVth degree σ. The difference between the first and second A section is on the last two bars. On the one hand, the last two bars of A_1 is another *turnaround* τ, and on the other hand, the last two bars of A_2 is an end slate on the tonic ω in anticipation of the bridge.
- Rule (iv) indicates that the final A section can be either A_1 or A_2.
- Rule (v) shows the composition of a B section in four 2-bar functions. Each δ represents respectively one key shift on the circle of fifth on degrees III VI II and V.
- The possible contents for each function are then learnt on the corpus. Each function is composed of a sequence of four chords with a duration of two beats each. Training them on a corpus enables the system to take many different possibilities into account for each function, and to constitute some form of style modelling for the generated chord progressions. The likelihood of each contents could be taken into account with probabilistic models [13] for a better emulation of the musical style. We trained the functions on the 'rhythm changes' sub-corpus from the Omnibook.

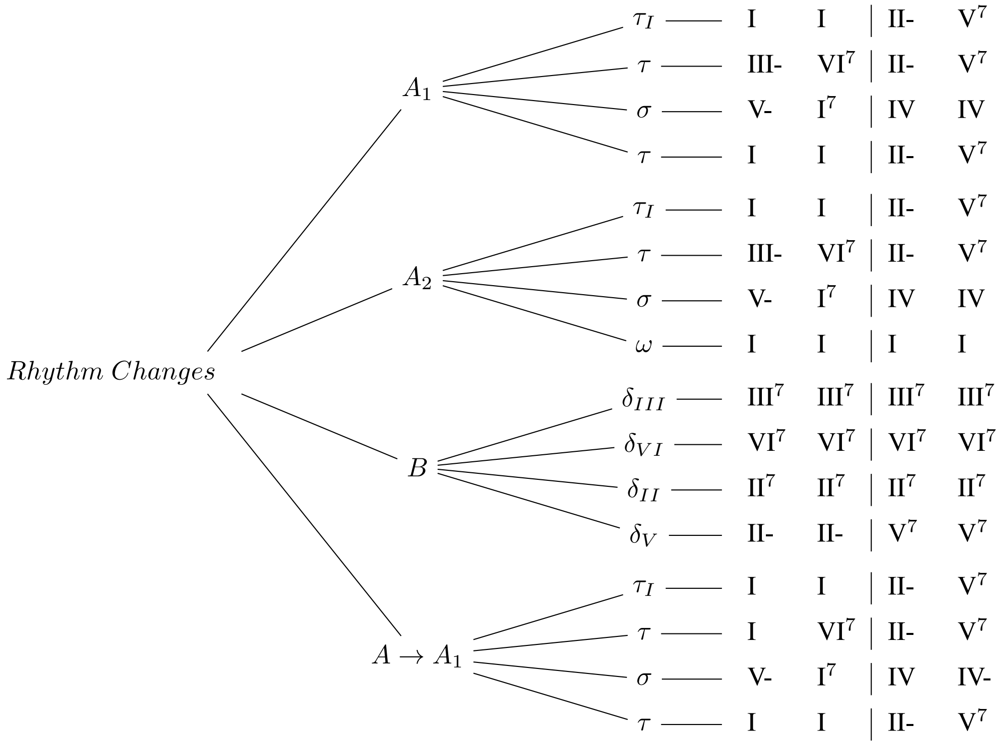

Rhythm Changes
A_1
τ_I — I I | II- V^7
τ — III- VI^7 | II- V^7
σ — V- I^7 | IV IV
τ — I I | II- V^7
A_2
τ_I — I I | II- V^7
τ — III- VI^7 | II- V^7
σ — V- I^7 | IV IV
ω — I I | I I
B
δ_{III} — III^7 III^7 | III^7 III^7
δ_{VI} — VI^7 VI^7 | VI^7 VI^7
δ_{II} — II^7 II^7 | II^7 II^7
δ_V — II- II- | V^7 V^7
$A \to A_1$
τ_I — I I | II- V^7
τ — I VI^7 | II- V^7
σ — V- I^7 | IV IV-
τ — I I | II- V^7

Figure 3. Diagram of a 'rhythm changes' derivation on Charlie Parker's theme *Celerity* (each chord lasts two beats). $A \to A_1$ denotes the application of rule (iv).

Figure 3 shows the diagram of one derivation of this grammar for one of the chord progressions on Charlie Parker's theme *Celerity*.

3. GENERATING MUSIC ON A MULTI-LEVEL PROGRESSION

In this section, we present a method using the information from a multi-level progression to enrich current music generation methods using the knowledge of the hierarchical structure and the equivalence between chord progressions. We first present the concepts we used as a basis for our work, and then introduce how to use the equivalent chord progressions of a multi-level progression in a music generation model and how it extends its possibilities.

3.1 Generating music on a chord progression

We propose to base our generation model on existing methods of machine improvisation guided by a scenario, to which we add the notion of multi-level progression.
The basis of our generation model is a simplified version of ImproteK [16] which is inspired by the work from OMax [32]. In OMax, improvising consists in navigating a memory on which we have information on the places sharing the same context, i.e. the same musical past. The fundamental premise is that it is possible to jump from one place in the memory to another with the same context. This non-linear path on the memory results in the creation of a new musical sentence that differs from the original material, but that retains its musical style [10]. ImproteK uses a similar approach, but with the addition of a temporal scenario used to guide the improvisation. The scenario is a prior known sequence of symbols (called *labels*) that must be followed during the improvisation. Contrary to Omax, the memory is not based on a sequence of pitches, but on a sequence of musical contents tagged by a label.

The goal of the associated generation model is to combine at every time in the generation an anticipation step ensuring continuity with the future of the scenario, and a navigation step keeping consistency with the past of the memory. Therefore, a phase of generation of a music sequence follows two succesive steps:

- The anticipation step consists in looking for an event in the memory sharing a common future with the current scenario. This is done by indexing the prefixes of the suffix of the current scenario in the memory using the regularites in the pattern [16]. Therefore, by finding such a prefix in the memory, we insure continuity with the future of the scenario.
- The navigation step consists in looking for events in the memory sharing a common context with what was last played in order to follow a non-linear path in the memory, thus creating new musical sentences.

As in OMax, the search for places in the memory sharing a common past is done thanks to the factor oracle: a structure from the field of formal language theory introduced by Crochemore et al. [7, 8]. Initially designed as an automaton recognising a language including at least all the factors of a given word, the factor oracle has been widely used in machine improvisation systems such as OMax, ImproteK, PyOracle [11] or CatOracle [12].

3.2 Considering a multi-level progression

We now present a method to take into account a multi-level progression, that is to say a scenario that is not just a sequence of symbols. If we want to keep the previous formalism, each label is now a list of symbols corresponding to each scale. We therefore need to propose a way to adapt the previous method to take this additional information into account in both the anticipation step and the navigation step.

- For the anticipation step, we first look for a prefix of the suffix of the current multi-level progression in the memory with exact labels; the contents of the memory must match the contents of the scenario on every scale. By doing so, we ensure the consistency of the musical contents in the global context of the scenario. If such a prefix cannot be found, we proceed to a search with equivalent labels. For instance, we can accept places in the memory without the right chord, but with the right function and the right section. We can favour places in the memory sharing a similiar multi-level label as the one in the multi-level progression according to a score (see below). The system can thus react and generate an improvisation on previously unmet chords, as long as they share a similar role in the scenario as previously known chords.

- The navigation step is done in a similar way. Using the information given by the factor oracle, we first look for places in the memory sharing the same context with exact labels, but we also open the possibilities to equivalent labels. This way, we extend the realm of possibilities for music generation. Once again, we can favour places in the memory sharing similar multi-level label as the one in the multi-level progression (with a weight); thus, taking better consideration of the progression's hierarchical structure.

In order to compute a score of similarity between multi-level labels, a weight W_i can be attributed to each level in order to evaluate the similarity of the considered places in the memory with the scenario such as

$$\sum_{i \in \text{level}} W_i = 1.$$

In this way, we can prioritise the choice of places in the memory with high scores, or limit the choice to highest scores according to a threshold. For instance, a user can consider that matching the functions is more important than matching the chords.

Finally, Figure 4 sums up the whole process for generating music upon multi-level progressions.

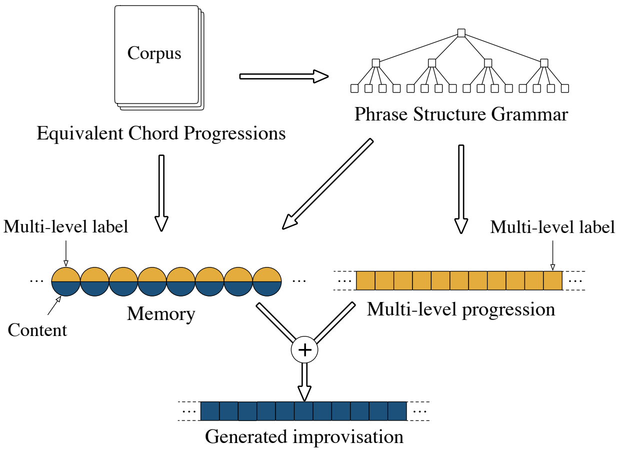


Figure 4. Process for generating an improvisation on a multi-level progression. First the phrase structure grammar is constructed with a musician using a corpus of equivalent chord progressions. Then the grammar is used to generate a multi-level progression, and used to provide information when creating the memory with multi-level labels. Finally, the improvisation is generated by navigating the memory guided by the multi-level progression.

I	II- V^7	I	II- V^7
I^7	IV^7	III- VI^7	II- V^7
I	II- V^7	III- VI^7	II- V^7
V- I^7	IV #IVo	I	I
III^7	III^7	VI^7	VI^7
VI-	II^7	II-	V^7
I	II- V^7	I VI^7	II- V^7
I I^7	IV #IVo	I	II- V^7

Figure 5. Example of derivation of 'rhythm changes' generated by the grammar.

4. EXPERIMENTATION ON RHYTHM CHANGES

4.1 Evaluation of the grammar

In order to evaluate our phrase structure grammar, we generated 30 derivations of 'rhythm changes' with it. The generated multi-level progressions have been assessed by the musician, who helped creating the grammar and by another professional jazzman, who was not involved in the initial process and therefore analysed the generated multi-level progressions strictly from a musicology point of view. Figure 5 shows an example of generated 'rhythm changes'. Other derivations of 'rhythm changes' can be found online at `repmus.ircam.fr/dyci2/ressources`.

Every generated 'rhythm changes' has been validated by both musicians. However, these generated chord progressions have to be seen only as realisations of 'rhythm changes' for improvisation accompaniment. For automatic composition of a chord progression for a theme, some par-

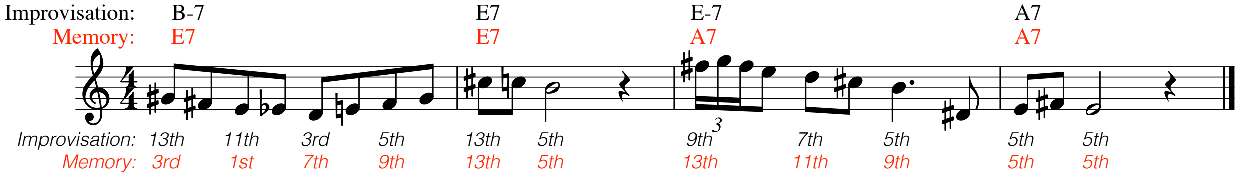

Improvisation: B-7 E7 E-7 A7
Memory: E7 E7 A7 A7
3
Improvisation: 13th 11th 3rd 5th 13th 5th 9th 7th 5th 5th 5th
Memory: 3rd 1st 7th 9th 13th 5th 13th 11th 9th 5th 5th

Figure 6. First four bars from the bridge of an improvisation on *An Oscar for Treadwell*. The 'Improvisation' lines shows the chord and degrees played in the generated improvisation; the 'Memory' lines shows the chord and degrees on which the different parts of the melody used to generate the improvisation were actually learnt. We see that when generating improvisations, we can reach places in the memory with a different chord label. The melody still makes sense in its continuity and harmonically because the multi-level labels from the memory and from the multi-level progressions are equivalent. The generated improvisations are therefore enriched with a new form of creativity.

allelism constraints assuring some symmetries between the different A section would be welcome. Moreover, this grammar covers the whole span of traditional bebop style 'rhythm changes' (on which it was trained on with the Omnibook corpus). No important possibilities were reported missing by any of the musicians. However, as expected, it does not create more modern versions of 'rhythm changes' such as the one of *The Eternal Triangle* by Sonny Stitt, or *Straight Ahead* by Lee Morgan, for instance. However, this is only due to the training corpus. No structural changes to the grammar would be needed for these. It would be interesting in the future to extend our 'rhythm changes' corpus to more varied chord progressions.

4.2 Improvisation on 'rhythm changes'

In order to test the benefits of using a multi-level progression, we generated some improvisations on Charlie Parker's music. First, using the phrase structure grammar trained on the 'rhythm changes' from the Omnibook, we generated multi-level progressions. On these progressions, we generated improvisations using two methods:

- the base generation model introduced in section 3.1. In this case, only the chord progression level of the multi-level progression was taken into account for generation.
- the extended generation model introduced in section 3.2. In this case, all the information from the multi-level progression was taken into account, i.e., the chord progression, the functional progression, and the sectional progression. A score was attributed to each level. We considered that for 'rhythm changes' the most important aspect was the functional aspect. Therefore, we attributed a weight of 0.5 to the functional level. We then attributed a weight of 0.3 to the chord level, and a weight of 0.2 to the sectional level.

In both cases, the content of the memory is a tune from Charlie Parker's Omnibook. Examples of generated improvisations (with both models) can be listened online at `repmus.ircam.fr/dyci2/ressources`.

We conducted a listening session with two professional jazz musicians to analyse the generated improvisations. First of all, the most significant difference seems to be that when using the multi-level progression, the improvisations are indeed better at following the structure. This can be especially noticed during the B section of the improvisations, where the guide notes and 5$^{\text{th}}$ of each chord are more pronounced (cf. example on *An Oscar for Treadwell*). Figure 6 shows an excerpt from the bridge of an improvisation on *An Oscar for Treadwell*.
Second, when encountering chords that do not appear in the memory, the freedom provided by the multi-level progression (e.g., playing on a different chord as long as it shares the same functional role) enables the improvisation system to generate musical sentences with less fragmentation, creating a better sense of consistence and fluidity (cf. example on *Thriving from a Riff*). Moreover, this generates a form of creativity exemplified by playing musical sentences on a different chord than the original one, thus playing new guide notes or extensions adding colour to the improvisations, whilst keeping consistency (cf. example on *Anthropology*).

These results are promising and show how using a multi-level progression can result in significant improvement on how the improvisations are guided through the memory of the system.

5. CONCLUSIONS

We have shown how we can model the multi-level progression of a chord progression with a phrase structure grammar, and how to use this information for machine improvisation. First, this abstraction of a chord progression enables the system to generate several instances (or voicings) of this chord progression injecting some creativity in the chord progression generation whilst keeping its multi-level structure under control. Second, this additional musical information is used during the generation of a melodic improvisation; the generation process is able to take into account the global form of the scenario with the multi-level aspect to adapt to an ever-changing scenario and expand its possibilities. We applied this method on 'rhythm changes'

and constructed a phrase structure grammar for this type of chord progressions. This grammar was built and validated with musicians expertise. We then generated improvisations on multi-level progressions generated with the grammar (including chord progression, functionnal progression and sectional progression), using new navigation heuristics adapted to this multi-level information. The generated improvisations give promising results according to professional jazz musicians. The global form of the chord progression is respected, and the improvisations are still consistent, even on unmet chord progressions.
The 'rhythm changes' grammar was trained on a corpus of traditional bebop style 'rhythm changes'. It would be interesting to extend this corpus to more modern variations of 'rhythm changes' and to apply this method on other scenarios such as a blues structure or more abstract scenarios such as Steve Coleman's Rhythmic Cycles based on the lunation cycle [33]. It would also be interesting to see if this hierarchical structure could be extended to a higher level to take the organisation of an improvisation upon several successive occurences of the chord progression into account. Finally, it would also be interesting to extend this work into a system where not only the improvisation adapts to the generated progression, but the derivations generated by the grammar also take what is being improvised into account, or into a system with an automatic generation of the hierarchical grammar based on machine learning on a corpus.

Acknowledgments

This work is made with the support of the French National Research Agency, in the framework of the project DYCI2 "Creative Dynamics of Improvised Interaction" (ANR-14-CE24-0002-01), and with the support of Region Lorraine. We thank the professional jazzmen Pascal Mabit and Louis Bourhis for their musical inputs.

6. REFERENCES

[1] S. Dubnov, G. Assayag, and R. El-Yaniv, "Universal classification applied to musical sequences," in *Proceedings of the International Computer Music Conference*, 1998, pp. 332–340.

[2] V. Padilla and D. Conklin, "Statistical generation of two-voice florid counterpoint," in *Proceedings of the 13th Sound and Music Computing Conference*, 2016, pp. 380–387.

[3] R. P. Whorley and D. Conklin, "Music generation from statistical models of harmony," *Journal of New Music Research*, vol. 45, pp. 160–183, 2016.

[4] F. Pachet, "The Continuator : musical interaction with style," in *Proceedings of the International Computer Music Conference*, 2002, pp. 211–218.

[5] M. I. Bellgard and C. P. Tsang, "Harmonizing music the boltzmann way," in *Musical Networks*, N. Griffith and P. M. Todd, Eds. MIT Press, 1999, pp. 261–277.

[6] G. Bickerman, S. Bosley, P. Swire, and R. M. Keller, "Learning to create jazz melodies using deep belief nets," in *Proceedings of the International Conference on Computational Creativity*, 2010, pp. 228–236.

[7] C. Allauzen, M. Crochemore, and M. Raffinot, "Factor oracle : a new structure for pattern matching," in *Proceedings of SOFSEM'99, Theory and Practice of Informatics*, 1999, pp. 291–306.

[8] A. Lefebvre and T. Lecroq, "Computing repeated factors with a factor oracle," in *Proceedings of the 11th Australasian Workshop On Combinatorial Algorithms*, 2000, pp. 145–158.

[9] G. Assayag and S. Dubnov, "Using factor oracles for machine improvisation," *Soft Computing*, vol. 8-9, pp. 604–610, 2004.

[10] G. Assayag and G. Bloch, "Navigating the oracle : a heuristic approach," in *Proceedings of the International Computer Music Conference*, 2007, pp. 405–412.

[11] G. Surges and S. Dubnov, "Feature selection and composition using PyOracle," in *Proceedings of the 2nd International Workshop on Musical Metacreation*, 2013, pp. 114–121.

[12] A. Einbond, D. Schwarz, R. Borghesi, and N. Schnell, "Introducing CatOracle: Corpus-based concatenative improvisation with the audio oracle algorithm," in *Proceedings of the 42nd International Computer Music Conference*, 2016, pp. 140–147.

[13] K. Déguernel, E. Vincent, and G. Assayag, "Using multidimensional sequences for improvisation in the OMax paradigm," in *Proceedings of the 13th Sound and Music Computing Conference*, 2016, pp. 117–122.

[14] D. Bailey, *Improvisation, its nature and practice in music*. Mootland Publishing, 1980.

[15] A. Donze, S. Libkind, S. A. Seshia, and D. Wessel, "Control improvisation with application to music," EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2013-183, November 2013.

[16] J. Nika, M. Chemillier, and G. Assayag, "ImproteK : introducing scenarios into human-computer music improvisation," *ACM Computers in Entertainment*, vol. 4, no. 2, 2017.

[17] J. Nika, "Guiding human-computer music improvisation: introducing authoring and control with temporal scenarios," Ph.D. dissertation, UPMC - Université Paris 6 Pierre et Marie Curie, 2016.

[18] S. Koelsch, M. Rohrmeier, R. Torrecuso, and S. Jentschke, "Processing of hierarchical syntactic structure in music," *Proceedings of the National Academy of Sciences*, vol. 110, no. 38, pp. 15 443–15 448, 2013.

[19] M. Mongeau and D. Sankoff, "Comparison of musical sequences," *Computer and the Humanities*, vol. 24, pp. 161–175, 1990.

[20] M. Giraud, R. Groult, E. Leguy, and F. Levé, "Computational fugue analysis," *Computer Music Journal*, vol. 39, no. 2, 2015.

[21] M. Giraud, K. Déguernel, and E. Cambouroupoulos, "Fragmentations with pitch, rhythm and parallelism constraints for variation matching," in *Proceedings of the 10th International Symposium of Computer Music Multidisciplinary Research*, 2014, pp. 298–312.

[22] D. Tymoczko, *A Geometry of Music: Harmony and Counterpoint in the Extended Common Practice*. Oxford University Press, 2011.

[23] E. Chew, *Mathematical and Computational Modeling of Tonality*. Springer, 2014.

[24] F. Lerdahl and R. Jackendoff, *A Generative Theory of Tonal Music*. MIT Press, 1983.

[25] N. Chomsky, *Studies on semantics in generative grammar*. Walter de Gruyter, 1996, vol. 107.

[26] W. B. de Haas, M. Rohrmeier, R. C. Veltkamp, and F. Wiering, "Modeling harmonic similarity using a generative grammar of tonal harmony," in *Proceedings of the 10th International Society for Music Information Retrieval Conference*, 2009, pp. 549–554.

[27] M. Steedman, "A generative grammar for jazz chord sequences," *Music Perception*, vol. 2, no. 1, pp. 52–77, 1984.

[28] M. Chemillier, "Toward a formal study of jazz chord sequences generated by Steedman's grammar," *Soft Computing*, vol. 9, no. 8, pp. 617–622, 2004.

[29] M. Levine, *The jazz theory book*. Sher Music, 1995.

[30] C. Parker and J. Aebersold, *Charlie Parker Omnibook*. Alfred Music Publishing, 1978.

[31] M. Kaliakatsos-Papakostas, D. Makris, A. Zacharakis, C. Tsougras, and E. Cambouropoulos, "Learning and blending harmonies in the context of a melodic harmonisation assistant," *Journal of Creative Music Systems*, vol. 1, no. 1, 2016.

[32] G. Assayag, G. Bloch, M. Chemillier, A. Cont, and S. Dubnov, "OMax Brothers : a dynamic topology of agents for improvization learning," in *Workshop on Audio and Music Computing for Multimedia*, 2006.

[33] S. Coleman, "The lunation cycle as a point of departure for musical ideas," in *Arcana II : Musicians on Music*, J. Zorn, Ed., 2007, pp. 56–61.

A CONSTRAINT-BASED FRAMEWORK TO MODEL HARMONY FOR ALGORITHMIC COMPOSITION

Torsten Anders
University of Bedfordshire
Torsten.Anders@beds.ac.uk

ABSTRACT

Music constraint systems provide a rule-based approach to composition. Existing systems allow users to constrain the harmony, but the constrainable harmonic information is restricted to pitches and intervals between pitches. More abstract analytical information such as chord or scale types, their root, scale degrees, enharmonic note representations, whether a note is the third or fifth of a chord and so forth are not supported. However, such information is important for modelling various music theories.

This research proposes a framework for modelling harmony at a high level of abstraction. It explicitly represents various analytical information to allow for complex theories of harmony. It is designed for efficient propagation-based constraint solvers. The framework supports the common 12-tone equal temperament, and arbitrary other equal temperaments. Users develop harmony models by applying user-defined constraints to its music representation.

Three examples demonstrate the expressive power of the framework: (1) an automatic melody harmonisation with a simple harmony model; (2) a more complex model implementing large parts of Schoenberg's tonal theory of harmony; and (3) a composition in extended tonality. Schoenberg's comprehensive theory of harmony has not been computationally modelled before, neither with constraints programming nor in any other way.

1. INTRODUCTION

In the field of algorithmic composition, harmony is a challenging area to address. Theories of harmony can be rather complex, as the mere size of standard harmony textbooks indicates. Also, different theories vary considerably depending on the musical style they address such as classical music [1], Jazz [2], contemporary classical music in extended tonality [3], or microtonal music [4].

As there is no agreement on a single theory of harmony, a flexible algorithmic composition environment should allow users to define their own theory. The present research provides a framework by which users can model their own theory of harmony, and then let the system generate music that follows it.

The framework provides building blocks common to many theories and that way simplifies the definition of custom theories from scratch. The proposed framework provides flexible representations of harmonic concepts (e.g., chords, scales, notes, as well as their parameters like a note's pitch or a chord root), which allow users to define their own harmony models declaratively at a high-level of abstraction with modular rules implemented by constraints that restrict the relations between these parameters.

Users can freely declare chord and scale types (e.g., major and minor triads and scales) by specifying pitch class intervals among chord or scale tones and their root. A number of different pitch representations are supported including pitch numbers, pitch classes, enharmonic note representations, scale degrees of notes and chords, and specific chord tones such as the fifth or the third of a triad. All these representations are freely constrainable. The framework supports the common 12-tone equal division of the octave (12-EDO) and arbitrary other equal divisions.

These models can generate harmonic progressions. If the harmonic rules are complemented by rules controlling the melody, counterpoint and so forth, then they can also be used more generally to generate music that follows given or generated progressions.

The presented framework is implemented in the music constraint system Strasheela [1] on top of the Oz programming language [5].

1.1 Plan of Paper

The rest of this paper is organised as follows. Section 2 puts this research into the context of previous work. Application examples of the proposed framework are shown in Sec. 3, while Sec. 4 presents formal details of the framework. Section 5 shows how to develop harmony models with this framework. The paper closes with a discussion that points out limitations of the proposal (Sec. 6).

2. BACKGROUND

Several algorithmic composition systems offer a rich representation of pitch-related and harmonic information, e.g., Impromtu, Opusmodus, the pattern language of SuperCollider, or music analysis systems like Humdrum and music21. However, such systems process this information procedurally, which makes it difficult to model the complex interdependencies found in harmony.

[1] http://strasheela.sourceforge.net/

The declarative approach of constraint programming [6] is more suitable for that. In this programming paradigm, users define *constraint satisfaction problems* (CSP) in which *constraints* restrict relations between *decision variables*. [2] Variables are unknowns, and initially have a *domain* of multiple possible values. A *solver* then searches for one or more *solutions* that reduces the domain of each variable to one value that is consistent with all its constraints.

Several surveys, partly by this author, give an overview of how constraint programming has been used for modelling music theory and composition. Pachet and Roy [7] focus on constraint-based harmonisation, specifically four-part harmonisation of a given melody. An introduction to the field of modelling composition and music theory with constraint programming in general is the detailed review [8]. A further review [9] presents how six composers used constraint programming to realise specific compositions.

A particular extensive constraint-based harmony system is CHORAL by Kemal Ebcioğlu [10], which generates four-part harmonisations for given choral melodies that resemble the style of Johann Sebastian Bach.

Several existing systems allow users to define their own harmonic CSPs, and each of these systems has its specific advantages. Situation [11] provides a family of mini-languages for specifying how the arguments to certain predefined harmonic constraints change across chords. Score-PMC, as subsystem of PWConstraints [12, Chap. 5] is well suited for complex polyphonic CSPs where the rhythm is fixed. PWMC [13] and its successor Cluster Engine allow for complex polyphonic CSPs where the rhythm is constrained as well.

While all these systems allow users to define their own harmonic CSPs, they restrict the constrainable harmonic information to pitches and intervals between pitches. For example, users can constrain harmonic and melodic intervals between note pitches, or represent an underlying harmony by chords consisting of concrete notes. However, the music representations of these systems are not extendable, and more abstract analytical information such as chord and scale types or roots, scale degrees, enharmonic note representations etc. are not supported. Such analytical information is important for various music theories. Some information could be deduced from a pitch-based representation (e.g., pitch classes), but other information is difficult to deduce (e.g., the chord type or root) and complex harmonic CSPs are more difficult to define and solve this way.

An expressive harmonic constraint system that supports such analytical information is the combination of the music representation MusES [14] and the constraint solver BackTalk [15]. This combination was used, e.g., for automatic harmonisation. BackTalk supports variable domains of arbitrary SmallTalk collections. The search algorithm first filters all variable domains with a general consistency-checking algorithm, and then searches for a solution of this reduced problem with a backjumping algorithm.

Many modern constraint solvers use constraint propagation for efficiency [16, Chap. 4]. Their variable domains are restricted to specific types (e.g., Boolean, integer and sets of integers), and highly optimised domain-specific propagation algorithms filter variable domains before every step in the search process, which typically greatly reduces the search space. The design of MusES cannot be ported to propagation-based constraint solvers, because it is not limited to the variable types supported by such solvers. MusES has originally not been designed with constraint programming in mind, and the variable domains of CSPs defined with MusES consist of complex SmallTalk objects.

This research proposes the first harmony framework that supports various analytical information to allow for modelling complex theories of harmony at a high level of abstraction, and whose design is at the same time suitable for propagation-based constraint solvers. Using constraint propagation allows, e.g., to quickly solve CSPs with large domain sizes and that way allows for microtonal harmony. While the framework was implemented in Strasheela on top of Oz, it could also be implemented with any other constraint system that supports the following features found in various propagation-based constraint systems (e.g., Gecode, Choco, JaCoP, MiniZinc): the variable domains Boolean, integer, and set of integers; integer and set constraints including reified constraints (meta-constraints, which also constrain whether their stated relationship holds or not); and the element constraint. [3]

This framework is largely comparable in terms of its flexibility with the combination of MusES and BackTalk, although it has a different stylistic focus. MusES was designed for jazz, while this research focuses on classical tonal music and contemporary music in an extended tonality including microtonal music.

3. APPLICATIONS

Before presenting formal details of the proposed framework, some examples showing the framework in action will further motivate this research. Please remember that all rules discussed for these examples are only a demonstration of the capabilities of the proposed framework, and all these rules can of course be changed.

3.1 Automatic Melody Harmonisation

The first example creates a harmonisation for a given melody. It is comparatively simple, and is therefore discussed in more detail.

While the proposed framework was originally developed for music composition, it can also be used for analysis, because it only controls relations between concepts like notes and chords. This example demonstrates an intermediate case. It performs an automatic harmonic analysis of a given folk tune, but additional compositional rules are applied to the resulting harmonies. Voicing is irrelevant in this example; only the chord symbols are searched for.

The harmonic rhythm is slower than the melody, as common is classical, folk and popular music. By contrast, most automatic harmonisation examples in the literature are choral-like with one chord per note.

[2] In the rest of the paper we just use the term *variable* for brevity.

[3] For details on the element constraint see `https://sofdem.github.io/gccat/gccat/Celement.html`, and also section 4.3.

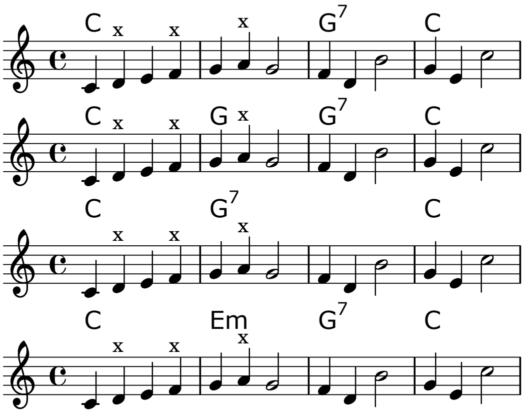


Figure 1. For the given melody this simple harmonisation model has four solutions

For simplicity, this example defines relatively rigid basic conditions. Only major, minor, and dominant seventh chords are permitted, and all chords must be diatonic in C major. The harmonic rhythm is fixed, and all chords share the same duration (e.g., a whole bar), but chord repetitions are permitted.

The example distinguishes between harmonic and nonharmonic tones, but for simplicity only a few cases of nonharmonic tones are permitted (passing and neighbour tones). All other melody notes must be chord pitches.

The example further borrows a few harmonic rules from Schoenberg [1] in order to ensure musically reasonable solutions. The example assumes that the given melody starts and ends with the tonic – these chords are constrained to be equal. A seventh chord must be resolved by a 'fourth upwards the fundament' (e.g., $V^7 \rightarrow I$), the simplest resolution form for seventh chords. All chords share at least one common pitch class with their predecessor (harmonic band, a simpler form of Schoenberg's directions for producing favourable chord progressions).

Figure 1 shows all solutions for the first phrase of the German folksong "Horch was kommt von draussen 'rein" that fulfil the given rules. An x on top of a note denotes a nonharmonic pitch.

Because of the relative simplicity of this example, it works only well for some melodies and less good for others. The harmonic rhythm of the melody must fit the harmonic rhythm specified for the example (at least chords can last longer, as repetitions are permitted). This is easily addressed by turning the currently fixed chord durations into variables, but doing so increases the size of the search space. Further, the nonharmonic pitches of the melody must fit the cases defined (passing and neighbour tones). An extension could define further cases, like suspensions and anticipations. Also, the melody currently cannot modulate. This can be solved by additionally modelling scales, and applying modulation constraints between chords and scales (as shown in the next example).

3.2 Modelling Schoenberg's Theory of Harmony

The next example implements large parts of Schoenberg's tonal theory of harmony [1] – a particular comprehensive theory – and that way demonstrates that the framework is capable of modelling complex conventional theories. To the knowledge of the author, Schoenberg's theory of harmony has not been computationally modelled before, neither with constraints programming nor in any other way. Also, this example implements modulation, which has rarely been done with constraint programming before. Among the many proposed systems, Ebcioğlu's CHORAL [10] is the only system the author is aware of that supports modulation. As the current example implements a lot of harmonic knowledge, it can only be summarised here.

The example modulates from C major to G major, but features an extended pitch set that allows for non-diatonic tones. Figure 2 shows a solution.

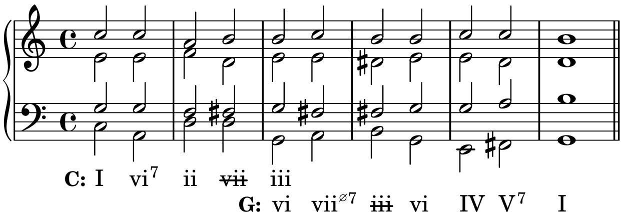


Figure 2. A solution of the Schoenberg harmony model modulating from C to G major; Roman numerals added manually for clarity (altered chords are crossed out)

The example applies Schoenberg's guidelines on chord root progressions designed to help obtaining better progressions. Schoenberg distinguishes between three kinds of root progressions. In *strong* or *ascending* progressions the chord root progresses by a fourth up (harmonically the same as a fifth down) or a third down. For example, in Fig. 2 chord 1 progresses to chord 2 by a third down ($I \rightarrow vi$), while chord 2 progresses to chord 3 by a fourth up / fifth down ($vi \rightarrow ii$). *Descending* root progressions form the second type. They are the inversion of strong progressions: a fourth down (fifth up) or a third up. These are not allowed in this example. Finally, in *superstrong* progressions the root moves a second up or down, as the chords in the penultimate bar do ($IV \rightarrow V$). These different progressions are explained in more detail and formalised in [17].

The chords are related to underlying scales, which change during the modulation. The first five chords relate to the C major scale, and the rest to G major. However, the example also allows for non-diatonic tones. Schoenberg introduces these as accidentals from church modes, namely the raised 1st, 2nd, 4th, 5th and the flattened 7th degree. These are always resolved stepwise in the direction of their alteration (e.g., raised degrees go up). In order to support the intended direction of the modulation (C to G), only raised degrees are allowed in this example. In the solution above, e.g., chord 4 contains a raised 4th degree of C major (F♯) that is chromatically introduced and resolves upwards by a step. Chord 7 contains a raised 5th degree of G major (D♯).

A modulation constraint requires that chord 5 is a *neutral chord*, i.e., a chord shared by both keys. In the solution above this chord is iii in C and vi in G major. The modulation constraint further requires that the neutral chord is followed by the *modulatory chord*, which is only part of the target key (vii$^{\varnothing 7}$ in the solution). For clarity, the modulatory chord must progress by a strong root progression a fourth upwards (into ~~iii~~ above).

The example also applies voice leading rules. Open and hidden parallel fifths and octaves are not permitted. Also, the upper three voices are restricted to small melodic intervals and small harmonic intervals between voices (larger harmonic intervals are allowed between tenor and bass).

Root positions and first inversions can occur freely. For example, chord 4 is a first inversion in Fig. 2. The number of first inversions has not been constrained and it is rather high in the shown solution (6 chords out of 11). More generally, statistical properties such as the likelihood of these chords are difficult to control by constraint programming (e.g., their overall number can be restricted, but they then may be bunched early or late in the search process).

It should be noted that the pitch resolution of this example is actually 31-tones per octave (31-EDO). This temperament is virtually the same as quarter-comma meantone, a common tuning in the 16th and 17th century. It can be notated with standard accidentals (♯, ♭, 𝄫 . . .). This temperament has been used here, because it simplifies the notation by distinguishing between enharmonically equivalent pitches (e.g., E♭ and D♯ are different pitch classes in this temperament). More generally, the use of this temperament demonstrates that the proposed framework supports microtonal music [18].

3.3 A Compositional Application in Extended Tonality

The last example discusses the 7 minute composition *Pfeifenspiel* by the author, which was composed for the two organs of the Kunst-Station St. Peter in Köln (premiered at the Computing Music VIII series in 2012). An excerpt from the piece is shown in Fig. 3 on the following page.

The music is tonal in the extended sense of Tymoczko [19]: melodic intervals tend to be small; the dissonance degree of the harmony is rather consistent; relatively consonant chords are used in moments of musical stability; sections of the piece are limited to certain scales; and for specific sections one tone is particularly important (root).

However, the piece is clearly non-diatonic. Suitable scales where found by first searching with an ad-hoc constraint program through about 200 scale and 50 chord types for scales that contain many chords with a similar dissonance degree (measured with an experimental algorithm). Solution scales were further evaluated manually by considering all chords that can be built on each scale degree, and by judging the melodic quality of scales. In the end, three scales that are all somewhat similar to the whole tone scale where selected: Takemitsu's *Tree Line* mode 2, Messiaen's mode 3, and Messiaen's mode 6. Two of these scales are shown in Fig. 3 in the analysis in the lowest stave (e.g., Takemitsu's *Tree Line* mode 2 on D in measures 6–7).

Based on these scales, a global harmonic and formal plan was composed by hand, but concrete harmonic progressions were generated algorithmically with custom harmony models for different sections. Also, contrapuntal sections rendering the harmony were algorithmically generated (and slightly manually revised), while some other sections were composed manually (e.g., in Fig. 3 the septuplets in the Great division, and the triplets in the pedal were composed manually).

Some example constraints are outlined. Chords have at least four tones, which all belong to the simultaneous scale. The first and last chord root of a section is often the root of its scale. To ensure smooth transitions between chords, the voice-leading distance between consecutive chords is low (at most 3 semitones in the excerpt). The voice-leading distance is the minimal sum of absolute intervals between the tones of two chords. For example, the voice-leading distance between the C and A♭ major triads is 2 (C $\rightarrow$ C $= 0$, E $\rightarrow$ E♭ $= 1$, G $\rightarrow$ A♭ $= 1$). Also, any three consecutive chords must be distinct.

The actual notes (in staves 1-3) must express the underlying harmony (stave 4). Nonharmonic tones (marked with an x in Fig. 3) are prepared and resolved by small intervals. Across the section starting in measure 8, the contrapuntal lines in the swell division rise gradually (pitch domain boundaries are rising), and melodic intervals are getting smaller (this section lasts over 10 bars, so this is not obvious from the few bars shown). The contrapuntal voices are never more than an octave apart; they don't cross; they avoid open and hidden parallels; they avoid perfect consonances between simultaneous notes (one is there in Fig. 3 after manual revisions); and voice notes sound all tones of the underlying harmony. Also, the lines are composed from motifs; and durational accents are constrained [20].

4. THE FRAMEWORK

This section describes formal details of the proposed framework. It explains how musical objects (notes, chords, and scales) are represented. Notes are concrete musical objects that produce sound when the score is played, but chord and scale objects represent analytical information – the underlying harmony.

Notes, chords and scales are represented by tuples of decision variables. When users define harmony models with this framework, they employ these objects and apply constraints between their variables. However, some wellformedness constraints must always hold between these variables, and these constraints are discussed here as well.

For clarity and portability, this section shows core definitions of the framework in mathematical notation instead of using any programming language (e.g., Oz). For simplicity, we leave out some auxiliary variables (intermediate results represented by extra variables).

4.1 Declaration of Chord and Scale Types

In the proposed framework, the chord and scale types supported globally by a constraint model (e.g., major and minor triads and scales) can be declared independently of the rest of the model. The ordered sequence $\mathcal{CT}$ consists of tuples, where each tuple specifies one chord type with a set of features as shown in the example below (1). The first tuple declares the major triad type: it specifies the pitch class integer representing the untransposed chord root (C), and the

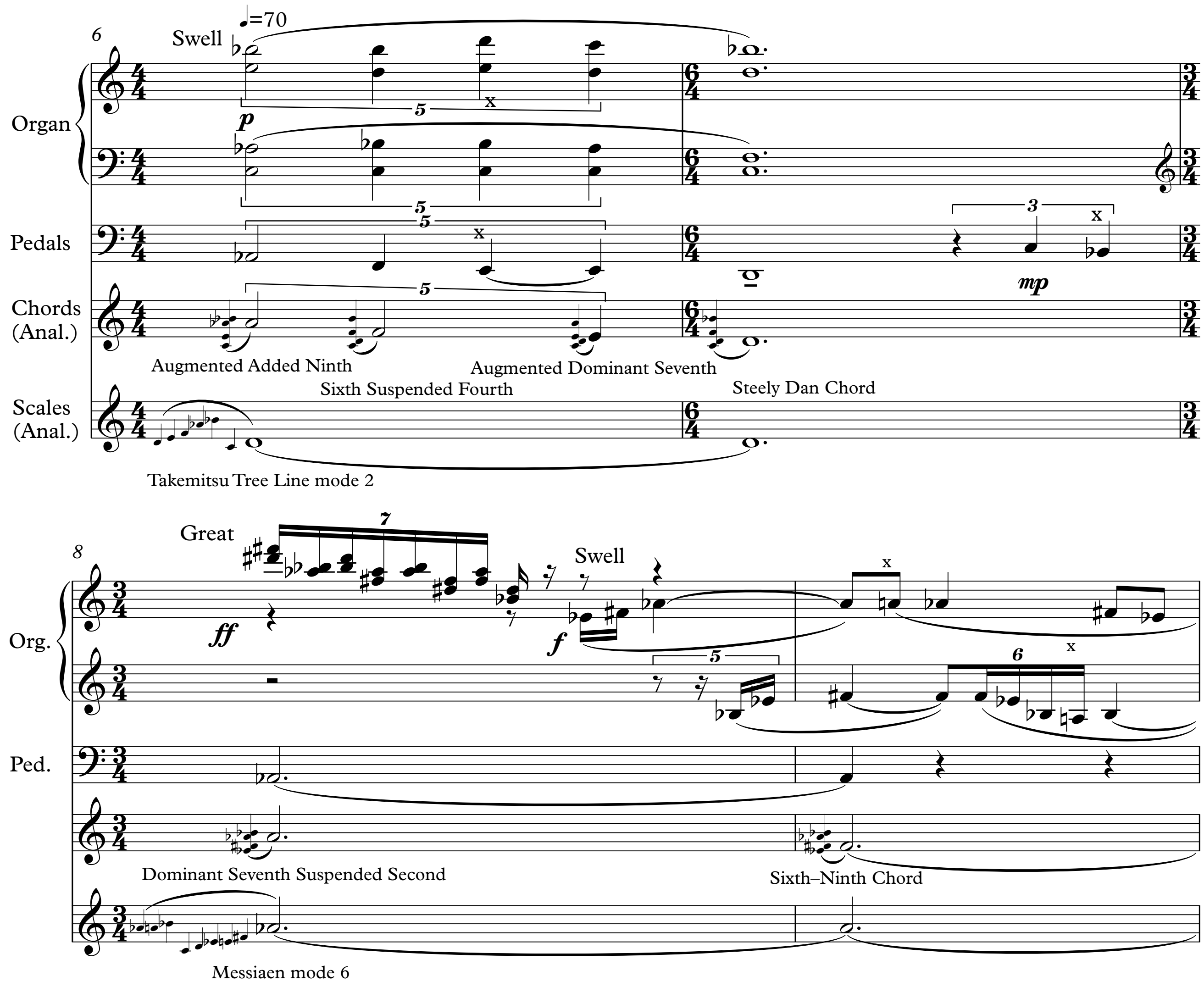


Figure 3. Excerpt from the composition *Pfeifenspiel* composed by the author. The upper three staves show the actual composition, and the lower two an analysis of the underlying harmony. Chord and scale tones are shown like an appoggiatura and roots as normal notes. Nonharmonic tones are marked by an x.

pitch classes of the untransposed chord – in this case the C-major triad, $\{\mathrm{C}, \mathrm{E}, \mathrm{G}\}$ – as a set of pitch class integers. The given $name$ is a useful annotation, but not directly used by the framework.

$$\mathcal{CT} = \begin{array}{l}[\langle name\text{: major}, \; root\text{: } 0, \; PCs\text{: } \{0,4,7\}\rangle, \\ \langle name\text{: minor}, \; root\text{: } 0, \; PCs\text{: } \{0,3,7\}\rangle, \\ \ldots]\end{array} \tag{1}$$

Scale types are declared in the same way in an extra sequence of tuples $\mathcal{ST}$. For example, the pitch class set of the major scale type is $\{0, 2, 4, 5, 7, 9, 11\}$, while its root is 0.

Users can also declare the global number of pitches per octave ($psPerOct$), and that way specify the meaning of all integers representing pitches and pitch classes in the constraint model.[4] A useful default value for $psPerOct$ is 12, which results in the common 12-EDO, and which was used for the chord and scale type examples above.

Instead of specifying pitch classes by integers as shown, it can be more convenient to specify note names, which are then automatically mapped to the corresponding pitch class integers, depending on $psPerOct$. In 12-EDO, C $\mapsto$ 0, C♯ $\mapsto$ 1 and so on. Alternatively, pitch classes can be specified by frequency ratios as a useful approximation of just intonation intervals for different temperaments. Again in 12-EDO, the prime $\frac{1}{1} \mapsto 0$, the fifth $\frac{3}{2} \mapsto 7$ etc.[5]

The format of chord and scale declarations is extendable. Users can add further chord or scale type features (e.g., a measure of the dissonance degree of each chord type), which would then result in further variables in the chord and scale representation discussed in the Sec. 4.3 below.

Note that chord and scale declarations are internally rearranged for the constraints discussed in Sec. 4.3. For example, $root_{\mathcal{CT}}$ is the sequence of all chord roots (in the order of the chord declarations), $PCs_{\mathcal{ST}}$ is the sequence of the pitch class sets of all scale declarations, and so on.

[4] Only equal temperaments that evenly subdivide the octave are supported. However, just intonation or irregular temperaments can be closely approximated by setting $psPerOct$ to a high value (e.g., $psPerOct = 1200$ results in cent resolution).

[5] Remember that for the frequency ratio r, the corresponding pitch class is $round((\log_2 r) \cdot psPerOct)$.

4.2 Temporal Music Representation

The underlying harmony can change over time. Temporal relations are a suitable way to express dependencies: all notes simultaneous to a certain chord or scale depend on that object (i.e., those notes fall into their harmony).

The framework shows its full potential when combined with a music representation where multiple events can happen simultaneously. A chord sequence (or scale sequence or both) can run in parallel to the actual score, as shown in the score example discussed above (Fig. 3).

Score objects are organised in time by hierarchic nesting. A sequential container implicitly constrains its contained objects (e.g., notes, chords, or other containers) to follow each other in time. The objects in a simultaneous container start at the same time (by default). All temporal score objects represent temporal parameters like their $start$ time, end time and $duration$ by integer variables. A rest before a score object is represented by its temporal parameter offset time (another integer variable), which allows for arbitrary rests between objects in a sequencial container, and before objects in a simultaneous container.

Equation (2) shows the constraints between temporal variables of a simultaneous container sim and its contained objects $\mathsf{object}_1 \ldots \mathsf{object}_n$. Any contained object – object_i – starts at the start time of the container sim plus the offset time of the contained object. The end time of sim is the maximum end time of any container. The relations between temporal variables of a sequential container and its contained objects are constrained correspondingly, and every temporal object is constrained by the obvious relation that the sum of its start time and duration is its end time. The interested reader is referred to [21, Chap. 5] for further details.

$$\begin{aligned} start_{\mathsf{object}_i} &= start_{\mathsf{sim}} + \mathit{offset}_{\mathsf{object}_i} \\ end_{\mathsf{sim}} &= \max(end_{\mathsf{object}_1}, \ldots, end_{\mathsf{object}_n}) \end{aligned} \quad (2)$$

Temporal relations can be defined with these temporal parameters. For example, we can constrain that (or whether, by using a resulting truth value) two objects o_1 and o_2 are simultaneous by constraining their start and end times (3). Note that for clarity this constraint is simplified here by leaving out the offset times of these objects, and remember that $\wedge$ denotes a conjunction (logical and).

$$start_{\mathsf{o}_1} < end_{\mathsf{o}_2} \wedge start_{\mathsf{o}_2} < end_{\mathsf{o}_1} \quad (3)$$

Remember that all these relations are constraints – relations that work either way. The temporal structure of a score can be unknown in the definition of a harmonic CSP. Users can apply constraints, e.g., to control the harmonic rhythm in their model, or the rhythm of the notes in a harmonic counterpoint.

If other constraints depend on which objects are simultaneous to each other (e.g., harmonic relations between notes and chords), then the search should find temporal parameters relatively early during the search process.

4.3 Chords and Scales

The proposed model represents the underlying harmony of music with chord and scale objects. This section introduces the representation of these objects, their variables, and the implicit constraints between these variables. The representation of chords and scales is identical, except that chords depend on the declaration of chord types $\mathcal{CT}$, and scales on scale types $\mathcal{ST}$ (see Sec. 4.1). Therefore, the rest of this subsection only discusses the definition of chords.

A chord c is represented by a tuple of four variables (4) – in addition to the temporal variables mentioned above (Sec. 4.2) that are indicated with "…".[6]

$$\mathsf{c} = \langle type,\ transp,\ PCs,\ root, \ldots \rangle \quad (4)$$

The $type$ (integer variable) denotes the chord type. Formally, it is the position of the respective chord in the collection of chord type declarations $\mathcal{CT}$, see Eq. (1) above. The $transp$ (integer variable) specifies how much the chord is transposed with respect to its declaration. PCs (set of integers variable) is the set of (transposed) pitch classes of the chord, and the $root$ (integer variable) is the (transposed) root pitch class.

For chords where the $root$ is 0 (C) in the declaration, $transp$ and $root$ are equal. In a simplified framework, the variable $transp$ could therefore be left out. However, sometimes it is more convenient to declare a chord where the root is not C (e.g., leaving a complex chord from the literature untransposed, or stating the pitch classes of a chord by fractions where the root is not $\frac{1}{1}$). Therefore this flexibility is retained here with the separate variables $transp$ and $root$.

The constrains (5) and (6) restrict the relation between the variables of any chord object c, and the collection of chord type declarations $\mathcal{CT}$. The element constraint is a key here. It accesses in an ordered sequence of variables a variable at a specific index, but the index is also a variable.[7] In (5), $root_{\mathcal{CT}}[type]$ is the untransposed root of the chord $type$. The (transposed) chord $root$ is that untransposed root – pitch-class transposed by the constraint $transp\text{-}pc$. Pitch class transposition in 12-EDO with modulus 12 is well known in the literature. The definition here uses $psPerOct$ as divisor to support arbitrary equal temperaments. A corresponding constraint for pitch class sets is expressed in (6).

$$root_{\mathsf{c}} = transp\text{-}pc(root_{\mathcal{CT}}[type_{\mathsf{c}}], transp_{\mathsf{c}}) \quad (5)$$

$$PCs_{\mathsf{c}} = transp\text{-}PCs(PCs_{\mathcal{CT}}[type_{\mathsf{c}}], transp_{\mathsf{c}}) \quad (6)$$

When chords are extended by further variables (e.g., a chord type specific dissonance degree) the chord declarations and chord object variables are simply linked by further element constraints (e.g., $feat_{\mathsf{c}} = feat_{\mathcal{CT}}[type_{\mathsf{c}}]$).

4.4 Notes with Analytical Information

Note objects represent the actual notes in the score. A note n is represented by a tuple of variables as shown in (7). As with chords, temporal variables are left out for simplicity, and are only indicated with "…".

[6] Internally, some additional auxiliary variables are used in the implementation, $untransposedRoot$ and $untransposedPitchClasses$.

[7] The element constraint is notated here like accessing an element in an array. $x = xs[i]$ constrains x to the element at position i in xs, where both x and i are integer or set variables, and xs is a sequence of such variables.

$$\mathsf{n} = \langle pitch,\, pc,\, oct,\, inChord?,\, inScale?, \ldots \rangle \tag{7}$$

The note's $pitch$ (integer variable) is essential for melodic constraints. It is a MIDI note number in case of 12-EDO. The pc (integer variable) represents the pitch class (chroma) independent of the oct (octave, integer variable) component, which is useful for harmonic constraints. The relation between a note's $pitch$, pc and oct is described by (8); the octave above middle C is 4.

$$pitch = pc + (oct + 1) \cdot psPerOct \tag{8}$$

The Boolean variable $inChord?$ indicates a harmonic or nonharmonic note, i.e., whether the pc of a note n is an element of the PCs of its simultaneous chord c, implemented with a reified set membership constraint (9). The Boolean variable $inScale?$ denotes equivalently whether notes are inside or outside their simultaneous scale.

$$inChord?_{\mathsf{n}} = pc_{\mathsf{n}} \in PCs_{\mathsf{c}} \tag{9}$$

4.5 Degrees, Accidentals, and Enharmonic Spelling

So far, the formal presentation used two common pitch representations: the single variable $pitch$ and the pair $\langle pc,$ $oct\rangle$; further pitch-related representations can be useful to denote scale degrees (including deviations from the scale), tones in a chord (and deviations), and to express enharmonic notation. These representations are closely related. They "split" the pc component into further representations, depending on a given scale or chord. These representations are only briefly summarised here; formal details are left out due to space limitations.

Enharmonic spelling is represented with the pair of integer variables $\langle nominal, accidental\rangle$, where $nominal$ represents one of the seven pitch nominals (C, D, E, ...) as integers: 1 means C, 2 means D, ..., and 7 means B.[8] The variable $accidental$ is an integer where 0 means ♮, and 1 means raising by the smallest step of the current equal temperament. In 12-EDO, 1 means ♯, -1 means ♭, and -2 is 𝄫 and so on.[9]

The representation of enharmonic spelling depends on the pitch classes of the C major scale: the $nominal$ 1 is a reference to the pitch class at the first scale degree of C-major, 2 refers to the second degree and so on. The same scheme can be used with any other scale to express scale degrees with the pair of integer variables $\langle scaleDegree,$ $scaleAccidental\rangle$. The variable $scaleDegree$ denotes basically the position in the pitch classes of a given scale. If the variable $scaleAccidental$ is 0 (♮), then the expressed pitch class is part of that scale. Otherwise, $scaleAccidental$ denotes how far the expressed pitch class deviates from the pitch class at $scaleDegree$. This representation has been used in the Schoenberg example described in Sec. 3.2 to constrain the raised scale degrees I, II, IV, and V.

This scheme can further be used for chords with the pair of integer variables $\langle chordDegree, chordAccidental\rangle$. The integer variable $chordDegree$ denotes a specific tone of a chord, e.g., its root, third, fifth etc.; it is the position of a chord tone in its chord type declaration in $\mathcal{CT}$, while $chordAccidental$ (integer variable) indicates whether and how much the note deviates from a chord tone (for nonharmonic tones). This representation was also used in the model of Schoenbergs theory of harmony discussed above to recognise dissonances that should be resolved (e.g., any seventh in a seventh chord).

5. MODELLING WITH THE FRAMEWORK

The proposed framework consists primarily of the constrainable harmony representation presented in the previous section. Developing concrete harmony models with this foundation is relatively straightforward: the variables in the representation are constrained. This section shows formal details of the first application shown in section 3.1.

For simplicity, this example only uses three chord types: major, minor, and the dominant seventh chord. These are declared exactly as shown previously in equation (1), where only the dominant seventh chord is added with the pitch classes $\{0, 4, 7, 10\}$ and root 0. The example declares scale types in the same way; only the major scale is needed.

The music representation of this example consists of a nested data structure. The top level is a simultaneous container, which contains three objects: a sequential container of notes; a sequential container of chords; and a scale. In the definition, the note pitches and temporal values are set to the melody (the folksong "Horch was kommt von draussen 'rein"), the scale is set to C major, and all chord durations are set to whole note values. In other words, only the chord types and transpositions, and hence also their pitch classes and roots are unknown in the definition.

Space does not permit to discuss all constraints of this example, but the formalisation of some of them gives an idea of the overall approach. As discussed before, the harmony of notes are their simultaneous chords and scale; chords also depend on their simultaneous scale. Only diatonic chords are allowed: every chord pitch class set is a subset of the scale pitch class set.

However, the example allows for nonharmonic notes. Constraining the notes' parameter $inChord?$ allows to model these. Nonharmonic notes are surrounded by harmonic notes, i.e., the parameter $inChord?$ of their predecessor and successor notes must be true. Only passing tones and neighbour tones are permitted: the melodic intervals between a nonharmonic note and its predecessor and successor notes must not exceed a step (two semitones). Equation (10) shows these constraints; remember that the operator ⇒ indicates the implication constraint (logical consequence).

$$\begin{aligned}
&inChord?_{\mathsf{n}_i} = \text{false} \Rightarrow \\
&\qquad inChord?_{\mathsf{n}_{i-1}} = inChord?_{\mathsf{n}_{i+1}} = \text{true} \\
&\quad \wedge |pitch_{\mathsf{n}_i} - pitch_{\mathsf{n}_{i-1}}| \leq 2 \\
&\quad \wedge |pitch_{\mathsf{n}_{i+1}} - pitch_{\mathsf{n}_i}| \leq 2
\end{aligned} \tag{10}$$

[8] The choice to start with C and not A as 1 is arbitrary, and it is very easy to change that when desired. C is represented by 1 and not 0 for consistency with scale degrees, where the lowest degree is commonly notated as I.

[9] This representation can also be implemented in a constraint system that only supports positive integer variables by adding a constant offset to all accidental variables.

The complete constraint problem definition is then given to the solver, which returns one or more solutions.

6. DISCUSSION

The applications shown in this paper demonstrate that the proposed framework is rather flexible. Complex theories of harmony in different styles can be modelled; in composition applications, generated notes can depend on an underlying harmony by ensuring a suitable treatment of nonharmonic tones; and in an analysis, generated chords can depend on given notes.

However, the proposed design is best suited for tonal music. For example, any atonal pitch class set can be declared as well, but then information like the chord root is redundant (it can simply be ignored in a model, though).

While the framework also supports tonal music in an extended sense (see Sec. 3.3) and microtonal music, it is less suitable for spectral music composition. Spectral music is based on absolute frequencies (and their intervals) translated into pitches. This approach preserves the octave of each pitch and that way the order of pitches in a chord. By contrast, in the proposed model chord and scale types are expressed by pitch classes. Individual chord or scale pitches can thus be freely octave transposed while retaining the chord or scale identity. Such an approach allows to control melodic and harmonic aspects independently with constraints.

The proposed model could be changed to better support spectral music by expressing chords with absolute pitches instead of pitch classes, and by disregarding all information based on pitch classes (chord roots, scale degrees etc.), but then tonal music theories depending on such analytical information that is independent of an octave component cannot be modelled anymore. The music constraint system PWMC [13] and its successor Cluster Engine implement such an approach.

A compromise could be special rules that constrain specific chord tones – e.g., tones at or above a certain *chordDegree* – into or above certain octaves, or above other chord tones, like some popular music voicing recommendations do (e.g., in a $V^{7\sharp9}$ chord, the augmented ninth is preferred above the major third).

The framework supports microtonal music, but only equal divisions of the octave. Specifically, just intonation intervals are best represented by ratios, and unequal temperaments with floats, but the proposed framework only uses integers, because constraint propagation works very efficiently for those. Nevertheless, just intonation intervals can be closely approximated (see footnote 4).

In summary, this paper presents a framework for modelling harmony with constraint programming that is suitable for modern propagation-based constraint solvers. Its representation of harmonic concepts such as chords, scales, and a variety of pitch representations lead to harmony CSPs with a high level of abstraction. Three applications demonstrated the range and complexity of harmonic CSPs that are made possible.

7. REFERENCES

[1] A. Schoenberg, *Theory of Harmony*. University of California Press, 1983.

[2] M. Levine, *The Jazz Theory Book*. Sher Music Co., 1995.

[3] V. Persichetti, *Twentieth-Century Harmony: Creative Aspects and Practice*. W. W. Norton & Company, 1961.

[4] D. B. Doty, *The Just Intonation Primer. An Introduction to the Theory and Practice of Just Intonation*, 3rd ed. San Francisco, CA: Just Intonation Network, 2002.

[5] P. v. Roy and S. Haridi, *Concepts, Techniques, and Models of Computer Programming*. Cambridge, MA: MIT Press, 2004.

[6] K. R. Apt, *Principles of Constraint Programming*. Cambridge: Cambridge University Press, 2003.

[7] F. Pachet and P. Roy, "Musical Harmonization with Constraints: A Survey," *Constraints Journal*, vol. 6, no. 1, pp. 7–19, 2001.

[8] T. Anders and E. R. Miranda, "Constraint Programming Systems for Modeling Music Theories and Composition," *ACM Computing Surveys*, vol. 43, no. 4, pp. 30:1–30:38, 2011.

[9] T. Anders, "Compositions Created with Constraint Programming," in *The Oxford Handbook of Algorithmic Music*, A. McLean and R. T. Dean, Eds. Oxford University Press, forthcoming.

[10] K. Ebcioğlu, "Report on the CHORAL Project: An Expert System for Harmonizing Four-Part Chorales," IBM, Thomas J. Watson Research Center, Tech. Rep. Report 12628, 1987.

[11] C. Rueda, M. Lindberg, M. Laurson, G. Block, and G. Assayag, "Integrating Constraint Programming in Visual Musical Composition Languages," in *ECAI 98 Workshop on Constraints for Artistic Applications*, Brighton, 1998.

[12] M. Laurson, "PATCHWORK: A Visual Programming Language and some Musical Applications," PhD thesis, Sibelius Academy, Helsinki, 1996.

[13] Ö. Sandred, "PWMC, a Constraint-Solving System for Generating Music Scores," *Computer Music Journal*, vol. 34, no. 2, pp. 8–24, 2010.

[14] F. Pachet, "An Object-Oriented Representation of Pitch-Classes, Intervals, Scales and Chords: The basic MusES," LAFORIA-IBP-CNRS, Universite Paris VI, Tech. Rep. 93/38, 1993, revised and extended version.

[15] P. Roy and F. Pachet, "Reifying Constraint Satisfaction in Smalltalk," *Journal of Object-Oriented Programming*, vol. 10, no. 4, pp. 43–51, 1997.

[16] F. Benhamou, N. Jussien, and B. O'Sullivan, Eds., *Trends in Constraint Programming*. London, UK: ISTE, 2013.

[17] T. Anders and E. R. Miranda, "A Computational Model that Generalises Schoenberg's Guidelines for Favourable Chord Progressions," in *6th Sound and Music Computing Conference*, Porto, Portugal, 2009, pp. 48–52.

[18] ——, "A Computational Model for Rule-Based Microtonal Music Theories and Composition," *Perspectives of New Music*, vol. 48, no. 2, pp. 47–77, 2010.

[19] D. Tymoczko, *A Geometry of Music: Harmony and Counterpoint in the Extended Common Practice*. Oxford: Oxford University Press, 2011.

[20] T. Anders, "Modelling Durational Accents for Computer-Aided Composition," in *Proceedings of the 9th Conference on Interdisciplinary Musicology CIM14*, Berlin, Germany, 2014, pp. 90–93.

[21] ——, "Composing Music by Composing Rules: Design and Usage of a Generic Music Constraint System," Ph.D. dissertation, School of Music & Sonic Arts, Queen's University Belfast, 2007.

pch2csd: an application for converting Nord Modular G2 patches into Csound code

Gleb Rogozinsky
The Bonch-Bruevich
Saint-Petersburg State
University of Telecommunications,
St. Petersburg, Russia
gleb.rogozinsky@gmail.com

Mihail Chesnokov
JSC SEC "Nuclear Physics Research"
St. Peterburg, Russia
chesnokov.inc@gmail.com

Eugene Cherny
Åbo Akademi University, Turku, Finland
ITMO University,
St. Peterburg, Russia
eugene.cherny@oscii.ru

ABSTRACT

The paper presents the pch2csd project, focused on converting patches of popular Clavia Nord Modular G2 synthesizer into code of Csound language. Now discontinued, Nord Modular G2 left a lot of interesting patches for sound synthesis and algorithmic composition. To give this heritage a new life, we created our project with the hope for being able to simulate the original sound and behavior of Nord Modular.

1. INTRODUCTION

Clavia Nord Modular was one of most remarkable hardware synthesizers of the late 90s. It inspired a whole new generation of modular synthesizer fans by combining the immediacy of dedicated hardware with the power and flexibility of computer-based programming. The suite of components comprising the NM project included an extensive list of modules (from oscillators to effects), an attractive software graphical interface and on the hardware side, keyboard and rack options. This synthesizer has been used extensively by many artists such as Astral Projection, Autechre, The Chemical Brothers, Somatic Responses, Junkie XL, Mouse on Mars, Nine Inch Nails and Covenant among others.

What really makes Nord Modular unique and sustains its relevance is the active community, which created a rich archive of patches. Alongside their inherent use to musicians, these patches can also incorporate a valuable educational aspect, inspiring people to develop their own creative signal processing and sequencing skills.

Unfortunately, Clavia ceased the Nord Modular production in 2009, and at present time it could be hard to buy the synth from the second-hand market to explore the vast collection of creative patches that were made by the community since the late 90s.

To facilitate the liberation of the NM patches from the closed source software and provide it with a continuing existence, we started a project called pch2csd in 2015 to (re)implement the Clavia Nord Modular G2 sound engine in Csound, a well-known sound and music computing system. Csound is one of the oldest computer music systems, it carries the sounds (and patches) from the past, which were originally compiled on mainframe computers back in the days, but have been continually re-compiled throughout the intervening years. Even now Csound can compile many sources written for the older versions of the language in the 80s. But despite it's heritage, at the present time Csound is being actively developed, and currently, the Csound code can interoperate with almost any programming language, providing robust, high performance and cross-platform solution for sound computing in real-time applications. Csound had been ported to iOS, Android and Raspberry Pi, there are Csound-based DAWs, tools to transform orchestra files to VST/VSTi plugins, a library for the Unity3D game engine [1], Jupyter notebook bindings and many more [1] .

The fist report on the pch2csd project was made at the Third International Csound Conference (St. Petersburg, Russia 2015) [2]. We presented the main concept of the project, the brief description of the code and a very simple "proof of a concept" working example. Since then, a number of improvements has been made, including new modules, mapping tables, and other code generation improvements.

This paper provides the report on current project status, as well as the engine implementation details.

The paper is organized as follows. Section 2 lists some other works on simulating music hardware in Csound. Section 3 provides the overview of Nord Modular G2 and it's patch format. Section 4 describes the implementation of the sound engine in Csound language, including the transformation from the NMG2 patch format. Section 5 discusses several patch conversion examples. Section 6 describes the limitations of the project. In the section 7 we conclude the paper and provide some directions for future developments.

2. RELATED WORK

Csound already has its own history of hardware emulation. There are several opcodes which emulate Moog filters and also the mini-Moog synthesiser. The DirectHam-

[1] A (not extensive) list of tools and instruments built on top of Csound can be found at the community websie, URL: http://csound.github.io/create.html

mond synthesiser written by Josep Comajuncosas emulates Hammond with the Leslie effect [3]. The Csound Book gives a simulation of the famous Roland TB303 [4]. Another example of TB303 was given by Iain McCurdy, who also wrote a code for emulation of Korg Mini Pops 7 and Roland TR-808 drum modules [5]. Csound had been used for emulating Yamaha DX7 [6]. Several experiments were carried to emulate the sound of Access Virus and Roland JP80xx, for example [7]. Eugenio Giordani and Alessandro Petrolati created a commercial app which emulated VCS3 by EMS [8].

The DX7 implementation stands apart from other examples, as it translates the synth's presets into a dynamically generated Csound code. Essentially the same approach is used for this project, but the code generation is more complex in our simulation.

3. NORD MODULAR OVERVIEW

Nord Modular and Nord Modular G2 were a unique hybrid software-hardware systems. They ran on several DSPs, but were controlled through a GUI using the software editor which is still available from the Clavia website [2]. Once (re)programmed, device can be run in a stand-alone hardware-only mode. For programming, the device should be connected to PC or Mac via USB cable. One can also find a freely downloadable copy of demo editor.

The Nord Modular system works only in real-time so the number of modules and voices are the most important parameter. Different modules cause different load. To optimize the performance several modules were used with almost identical functions, but with different operability. For example, *OscA* oscillator allows changing its wavetype on a flight, comparing to *OscD*, which uses the same wavetypes but demands the patch to be recompiled. There are two main parts of the patch: *voice* and *fx*. The *voice* part is a subject of polyphony, comparing to *fx* part which receives the mix of all voices. Obviously, the greater the number of voices played simultaneously, the lesser number of modules can be used before overrun.

Nord Modular G2 uses fixed sampling rate of 96 kHz for audio signals and 24 kHz for control rate signals (i.e. modulation). The overall number of modules is around 200, including numerous sound generators, envelopes, lfo, filters, random generators, mixers, delay units, switches, MIDI units, logics, waveshapers, sequencers and FXs.

There are 17 sound generators, most of which provide classical waveshapes with modulations. There are also noise generators, simple physical models and a built-in DX7 model. The Filter part consists of 14 filters, i.e. various LP, HP, BP and BR models, and also comb filter, formant filter and a 16-band vocoder. The section also includes three different equalizers of up to 3 bands. The Envelope section provides 9 envelope generators from simple Decay-only generator to multistage one. There are 16 units in a Mixer sections starting from one channel volume controller to 8 channel mixing unit. The FX part includes chorus, phaser, flanger, digitizer, frequency and pitch shifters, scratcher, reverb and a compressor. Delay units are included in the separate Delay section. There are 10 different units, from static one-tap delay to complex stereo delays. The Switch section contains various switches and dmux/mux units. The Logic includes common binary logic units, i.e. typical logical functions, flip-flops, pulse dividers and counters. There are also ADC and DAC converters. All waveshaping units are placed in the Shaper part. There are 7 of them, i.e. clip, drive, saturation, rectifier and wrapper units. The Level section relates to any modulation types. It also contains the envelope follower unit. The Random section includes 6 various random generators. The Sequencer part includes 5 sequencers of different kinds. All of sequencers can be linked in chain. There also MIDI, In/Out and Note sections. The detailed info on each unit can be found in official NM2 manual.

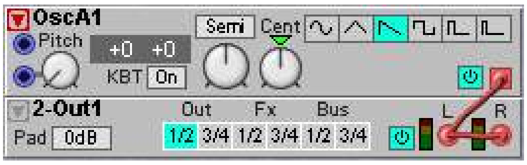


Figure 1. An example of the input-to-input connection: the output of the oscillator is connected to the both (left and right) inputs of the stereo output module.

A sound programmer connects the modules using virtual cables. The main cables types are: red (audio signals @ 96 kHz), blue (control signals @ 24 kHz), yellow (logic signals in the form of pulses @ 24 kHz), orange (logic signals @ 96 kHz).

In addition to these cable types there are also two user-defined types: green and violet. Actually the user can color any cable in any color using the context menu of an editor, i.e. change color of red cable into yellow. Also there are 'dead' cables of grey color. They appear if you break the valid link of modules.

Another peculiar feature of Nord Modular system is an ability to connect one input to another. So if you want to connect the output of a generator to both inputs of the *Out* module, it is allowed to connect one input to another (Fig. 1), and the output from the generator will be passed to both inputs.

Also, several modules can change their type, i.e. from red to blue, depending on input connections, i.e. connecting a red cable to any input of mixer turns it from default blue type to a red one.

Moreover, there are some hidden modules which were not included in the release version of the editor, i.e. Resonator, Driver, AR-Env, PolarFade. Most of them are not working, but some can be useful [3].

3.1 Patch format

Nord Modular G2 stores patches in a binary file with the extension *pch2*. The format is proprietary and has not been

[2] Nord Modular G2 official page, URL: http://www.nordkeyboards.com/downloads/legacy/nord-modular-g2

[3] A discussion about hidden modules in Nord Modular G2: http://www.electro-music.com/forum/viewtopic.php?t=54650

officially published by Clavia, so we rely on a community effort to decode the format done by Michael Dewberry in the 2005 [9]. According to the information he published, the *pch2* file starts with a text header terminated by the NULL byte. It contains general information about the patch (e.g. version, file type, etc.) in human readable form. This text header is followed by a two-byte binary header, representing the patch format version and the file type (a *patch* or a *performance*). All other information is stored in a series of "data objects". A byte layout of some objects is presented in the Appendix. For the purpose of this project we do not use such objects as Knob Assignments, MIDI Controller Assignments, Module Names and Textpad. Hence we read only those, representing the actual sound modules and their connections. For this, we first read Patch Description object (Table 3), then read two Module List objects (Table 4) for VA and FX [4], then skip the Mystery Object [5] (Table 1), then read two Cable List objects for VA and FX (Table 2), and finally we read two Module Parameters objects for VA and FX (Table 5).

4. IMPLEMENTATION

4.1 Overview

The project is implemented as a C program. The program extracts module and connection lists, as well as parameter values from the *pch2* file, and then composes a *csd* file from Csound code templates and mapping tables stored as plain-text files. We currently have 100 modules implemented as Csound user-defined opcodes (including filters, mixers, etc.) and 35 hand-crafted mapping tables to map values from the linear MIDI range to non-linear parameter range. Storing the data as text allows anyone to change the behavior of the modules without touching the C code.

4.2 Modeling NMG2 modules

We began with the challenges related to oscillators modeling. Any sound synthesis algorithm starts from some generator, thus the first and the most important part of the patch to be simulated is an oscillator section. The Nord oscillators produce aliased waveforms at 96 kHz. The Figure 2 shows the amplitude spectra of Clavia's sawtooth. The audio was recorded at 96 kHz/24 bit on the Focusrite Saffire PRO 40 audio interface. The solid line at the right border of plot is a Nyquist frequency (48 kHz). The spectra was calculated using 2048 points FFT with Hann window. We can clearly see the aliasing part of the spectra, mirrored from the Nyquist frequency. This feature of Nord Modular distinguishes it from the popular family of so-called analog-modeling synthesizers, which typically produce alias-free waveforms, and makes it possible to simulate the corresponding waves by simple generation of ideal piece-wise functions. Numerous records of the oscillator waveforms also prove it well.

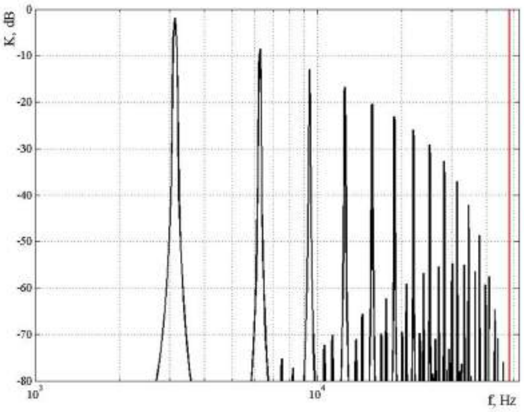


Figure 2. An amplitude spectra demonstrates the aliasing in the NMG2 sawtooth oscillator waveform. A solid vertical line to the right is the Nyquist frequency.

Filter analysis has been performed through connecting white noise generator to the corresponding filter of Clavia and comparing the output amplitude response with the filter models created by the authors.

4.3 Csound implementation details

To simulate the sound and behavior of modules of Clavia Nord Modular G2 we used Csound [10]. It is an open-source language for computer music and digital audio processing developed in MIT 1986 by Barry Vercoe and recently developed by a global community of enthusiasts. Csound can be run on all popular platforms including Android and iOS. Through Csound API it can be embedded in lots of applications, which makes it a nice candidate for Nord simulation together with a great amount of included opcodes [6] .

Csound document typically consists of two sections. The *Instruments* section contains all definitions of instruments to be used, and the *Score* part contains timeline events and function tables. Each of Nord modules is modeled as an User-Defined Opcode (aka UDO). At the conversion from pch2 to Csound the UDOs, which relate to patch's modules are read from txt files and included to the Csound document. Csound compiler reads code lines from top to the bottom, which is completely different approach to graphical system of patching, like MAX, Pd or Nord. Fortunately, Csound has a special patching system called *zak-space*. It provides a given number of audio rate (a-rate) and control rate (k-rate) buses to intercommunicate between different instruments and UDOs. There are separate commutation matrix for audio rate signals and another one for control rate signals. They are completely independent from each other, comparing to Nord patching system, where the cables are sequentially numbered. A separate part of our code solves that disparity by renumbering the cables after reading their numbers from the patch file. Comparing to typical behavior of Csound opcodes, where each opcode

[4] Note, that objects like this appear twice in the patch file, in the Voice Area and in the FX Area.

[5] We do not currently know the purpose of this object, but in our project we are only interested in a list of modules and their connections to reconstruct patches in Csound, and we have not yet found any cases where we would need to inspect this object more closely.

[6] All up-to-date Csound features can be found on the community website: `http://csound.github.io`

typically accepts an input data as some parameters and outputs the result, our zak-based UDOs do not have any outputs.

In more practical view, the typical Csound opcode/UDO is applied like this:

```
aOut1 SomeOpcode aIn1, aP1, kP2 [, ...]
```

where *aIn1* is an input of audio type, *aP1* is an a-rate parameter, *kP2* is a k-rate parameter, and *aOut1* is an a-rate output.

In our system we have

```
SomeOpcode aP1, kP2 [, ...], kIn1, kOut1
```

where *aP1* is an a-rate parameter, *kP2* is a k-rate parameter, *kIn1* is a number of some k- or a-rate bus to read input data from, and *kOut1* is a number of some k- or a-rate bus to send data to.

After parameter field our UDOs have IO field, in which the numbers of buses in zak space are listed. Using described approach we are free to list the opcodes in any order, just like Nord Modular user can add the modules in arbitrary order. So the first indexed module can be easily the last one in the audio chain.

Another important aspect to be described here is mapping. Nord modules have several different controllers of a lot of ranges, i.e. audio frequency range, amplitude range, normalized values in the range from 0 to 1, delay time ranges, envelope stage durations, etc. The real values of the controllers can be seen only when using editor. The patch file stores 7 bit MIDI values without any reference to appropriate range. It made us to manually fill the table with data types, ranges and values. Special mapping files of pch2csd project contain numbers of tables to be used for each parameter of each module.

I.e Module #112 (*LevAdd* module) table contains following lines:

```
s 2 LVLpos LVLlev
d BUT002
```

It means that the mapping table for the first parameter of *LevAdd* (value to add to input) is dependent on a second (*2*) parameter, which is the two-state button (table *BUT002*). The button switches *LevAdd* from unipolar to bipolar range of values. The mapping tables are placed in a separate subdirectory of a project.

Also during the development we had to solve a polymorphism issue. Several Clavia modules are polymorphous. Unfortunately there is no direct indication of current module type (a-rate or k-rate). It can be discovered only through analyzing cable connections. So our algorithm checks the module type, and its input connections. In case of non-default type of the input, the corresponding module twin is used instead of the default one.

5. EXAMPLES

Here we demonstrate the conversion of a real Clavia's patch (see picture ...). It consists of a noise generator, a simple 1-pole LP filter and an Output module. We give this extremely primitive patch not for the purposes of timbre discussion but rather to present a clear example of how our converter works.

The converter default output file is *test.csd* in the program directory. Comments are given after semicolons.

The algorithm detects three different types of modules and take their UDO definitions from the library.

```
sr  = 96000  ; audio sampling rate
ksmps = 16   ; times k-rate lower than a-rate
nchnls = 2   ; number of output channels
0dbfs = 1.0  ; relative amplitude level

zakinit 4, 3 ; Init zak-space

opcode Noise, 0, kkk ;White Noise generator
kColor, kMute, kOut  xin
if kMute!=0 goto Mute ;mutes sound if Mute is On
aout rand 0.5, 0.1 ; seed value 0.1
aout tone aout, kColor ; Csound simple LP filter
zaw aout, kOut
Mute:
endop

opcode FltLP, 0, kikkkkk ;One-pole LP filter
; Keyboard Tracking has not been implemented yet
kKBT, iOrder, kMod, kCF, kIn, kModIn, kOut xin
ain zar kIn
kmod zkr kModIn
aout tonex ain, kCF+kmod*kMod, iOrder
zaw aout, kOut
endop

opcode Out2, 0, kkkkk; Output module
; Only stereo output has been implemented
kTarget, kMute, kPad, kL, kR xin
if kMute!=0 goto Mute:
aL zar kL
aR zar kR
outs aL*kPad, aR*kPad
Mute:
endop

opcode Constant, 0, kk   ;Constant value
    ; Only bipolar mode now
kVal xin
zaw kVal, kOut ; CHANGE
endop

instr 1; VA section
    Noise 10000,0,2
    FltLP 0,1,0.5,1050,2,2,3
    Constant 24,2
    Out2 0,0,1,3,0
endin

instr 2; FX section
endin

; Here goes the score part of Csound.
; We just let it run for a long time...
i1 0 [60*60*24*7]
i2 0 [60*60*24*7]
```

We use two special buses per each matrix. Buses #0 contain silence. They connected to inputs without cables. Buses #1 are trash collectors. If output is not connected, it goes there.

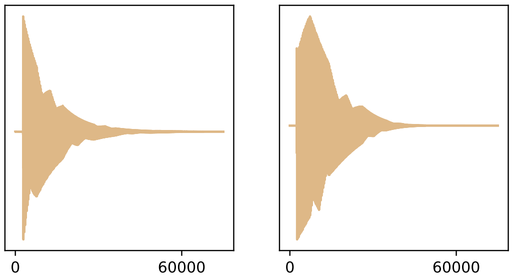


Figure 3. Sawtooth waveform's amplitude modulated with an envelope. Nord Modular G2 is on the left, our implementation is on the right.

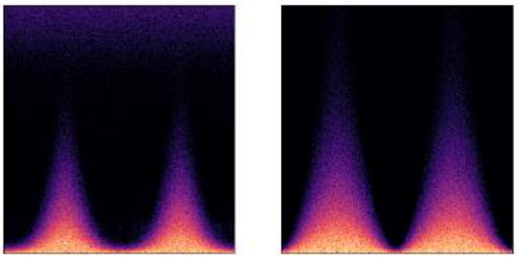


Figure 4. White noise filtered with LFO modulated lowpass filter. The file lengths are equal to 3 seconds. Nord Modular G2 (left) compared with our implementation (right). The Clavia's LFO is not exactly a sine.

6. EVALUATION

The practical study of the software shows several issues. Beginning from the oscillator block of the synthesis process, we discovered that Clavia oscillators run at arbitrary initial phase. So at each run or patch re-compilation the composite waveform of the sum of oscillators slightly differs. Also the exponential form of Clavia envelope generators seems to be sharper than real exp function 3. We also discovered that typical sine LFOs actually produce a bit asymmetric function. The figure 4 shows the difference between spectrograms of signals at the output of modulated LPF.

During developing we estimated several main limitations. First is related to the modules with non-determinate behaviour, i.e. all modules from the Random group. Each of those modules generates a random sequence, in which the current value depends on previous one. Such a behavior is not typical for Csound random generators. Another limitation relates to the patch presets. Each of Nord Modular patch contains 8 presets called Variations. User can select the desired preset from the interface. This feature has not been implemented yet. Despite listed limitations, we hope to completely overcome them in our future work.

Certainly, the in-depth evaluation of such project should include much more aspects than listed. Meanwhile, at the present stage of developing we are concentrated mostly on creating simulation which should be close to the original sound of NM2.

7. CONCLUSIONS AND FUTURE WORK

At its present state, the pch2csd project is ready for further development by international computer music enthusiasts. The core has been test to work on Windows and OSX systems. The user is able to open Clavia's patch file format pch2. The conversion log shows status of all dependencies, i.e. Csound UDOs, mapping tables, etc. It allows user to update the dependencies and also create his or her own versions of UDOs and mappings if needed. The module completion status is far from uniform. Most of straightforward modules are finished, i.e. mixing operations or switching. Random section seems most difficult, because of its ambiguous behavior. The same should be reported on FX section, although the actual quality of the reverb or chorus modules does not correlate with algorithmic patch behavior, which makes it not so critical.

Another important goal we started working on recently is to make the project hackable, so users would be able to easily modify module implementations to contribute to the project or to modify sound for their own tastes. The plain text modules and mapping tables were made as well as for that purpose, but the user experience here is still poor. As a first step to mitigate this we plan to develop a simple Electron-based UI[7] with an embedded text editor in the near future.

Our next to-do after providing a completely working solution is an integration with some existing Clavia patch editor. It will actually establish the new Clavia-based software modular system running on a Csound core. Also, the native Csound developments, i.e. Cabbage Studio[8] (a graphical UI for Csound build by Rory Walsh) seem very promising in the context of further integration.

Current tool sources can be found on the GitHub[9]: `https://github.com/gleb812/pch2csd`.

Acknowledgments

This work has been partially financially supported by the Government of the Russian Federation, Grant #074-U01.

8. REFERENCES

[1] R. Walsh, "Csound and unity3d," in *Proceedings of the Third International Csound Conference*. St. Petersburg, Russia: The Bonch-Bruevich St. Petersburg State University of Telecommunications, Mar. 2016, pp. 141–155. [Online]. Available: https://doi.org/10.5281/zenodo.50369

[2] G. G. Rogozinsky and M. Chesnokov, "Clavia nord modular g2 patch converter project," in *Proceedings of the Third International Csound Conference*. St. Petersburg, Russia: The Bonch-Bruevich St. Petersburg State University of Telecommunica-

[7] URL: `https://electron.atom.io/`

[8] URL: `https://csound.github.io/frontends.html#cabbage-and-cabbage-studio`

[9] The code, as well as sound examples, mentioned in the previous section, are also preserved at Zenodo, URL: `https://zenodo.org/record/581204`

tions, Mar. 2016, pp. 83–89. [Online]. Available: https://doi.org/10.5281/zenodo.50365

[3] Josep M Comajuncosas, "DirectHammond v1/v2 - csound instruments page." [Online]. Available: http://www.csounds.com/jmc/Instruments/instruments.htm#hammond

[4] R. C. Boulanger, *The Csound book: perspectives in software synthesis, sound design, signal processing, and programming*. MIT press, 2000.

[5] Iain McCurdy, "Csound realtime examples." [Online]. Available: http://iainmccurdy.org/csound.html

[6] "Csound DX7 patch translator." [Online]. Available: http://www.parnasse.com/dx72csnd.shtml

[7] G. G. Rogozinsky, E. Cherny, and I. Osipenko, "Making mainstream synthesizers with csound," in *Proceedings of the Third International Csound Conference*, G. G. Rogozinsky, Ed. The Bonch-Bruevich St. Petersburg State University of Telecommunications, 2016, pp. 132–140. [Online]. Available: http://dx.doi.org/10.5281/zenodo.50364

[8] E. Giordani and A. Petrolati, "Csound synthesis approach for the ivcs3 ios app," in *Proceedings of the Third International Csound Conference*. St. Petersburg, Russia: The Bonch-Bruevich St. Petersburg State University of Telecommunications, Mar. 2016, pp. 116–131. [Online]. Available: https://doi.org/10.5281/zenodo.50358

[9] M. Dewberry, "Nord Modular G2 Patch Format," https://web.archive.org/web/20160407003110/http://www.dewb.org/g2/pch2format.html/, 2005, [Online; accessed 03-Mar-2017].

[10] V. Lazzarini, S. Yi, J. ffitch, J. Heintz, . Brandtsegg, and I. McCurdy, *Csound - A Sound and Music Computing System*. Springer International Publishing, 2016, DOI: 10.1007/978-3-319-45370-5. [Online]. Available: http://link.springer.com/10.1007/978-3-319-45370-5

Appendix: NMG2 binary patch format

Field name	Value / Comment	Bits
Header byte	0x69	8
Length	0x4a	16
	Unknown	

Table 1. Mystery object

Field name	Value / Comment	Bits
Header byte	0x52	8
Location	0 / 1: FX / Voice	2
Unknown	-	14
Cable count	Nr. of cables in the area	8
Then, for each cable according to cable count:		
Color	0-6	3
Module from		8
Jack from	-	6
Type	0: in-to-in, 1: out-to-in	1
Module to	-	8
Jack to	-	6
	End of iteration	

Table 2. Cable list binary encoding

Field name	Value / Comment	Bits
Header byte	0x21	8
Length	-	16
Unknown	-	12
Voice Count	-	5
Height of FX/VA bar	-	14
Unknown	-	3
Red cable visibility	0: off, 1: on	1
Blue cable visibility	0: off, 1: on	1
Yellow cable visibility	0: off, 1: on	1
Orange cable visibility	0: off, 1: on	1
Green cable visibility	0: off, 1: on	1
Purple cable visibility	0: off, 1: on	1
White cable visibility	0: off, 1: on	1
Mono/Poly	-	2
Active variation	0-7	8
Category	0: No Cat, 1: Acoustic, 2: Sequencer, 3: Bass, 4: Classic, 5: Drum, 6: Fantasy, 7: FX, 8: Lead, 9: Organ, 10: Pad, 11: Piano, 12: Synth, 13: Audio In, 14: User 1, 15: User 2	8
Padding		-

Table 3. Patch description binary encoding

Field name	Value / Comment	Bits
Header byte	0x4a	8
Location	0: FX Area, 1: Voice Area	2
Module Count	Nr. of modules in the area	8
Then, for each module according to module count:		
Module type		8
Module index	Module ID	8
Horiz. position	-	7
Vert. position	-	7
Color	-	8
Appendix	-	4
If Appendix != 0:		
Unknown	-	2
Hidden parameter	-	4
End of iteration		
Padding		-

Table 4. Module list binary encoding

Field name	Value / Comment	Bits
Header byte	0x52	8
Length		16
Location	0 / 1: FX / Voice	2
Module count	-	8
For each module:		
Module index	-	8
Param. count	Nr. of parameters	8
For each parameter:		
Variation	0	8
Value	7-bit MIDI value	7
Variation	1	8
	. . .	
Variation	8	8
Value	7-bit MIDI value	7
End of both iterations		
Padding	8	8

Table 5. Module parameters binary encoding

Towards an Automated Multitrack Mixing Tool using Answer Set Programming

Flavio Everardo
Potsdam University
flavio.everardo@cs.uni-potsdam.de

ABSTRACT

The use of Answer Set Programming (ASP) inside musical domains has been demonstrated specially in composition, but so far it hasn't overtaken engineering areas in studio-music (post) production such as multitrack mixing for stereo imaging. This article aims to demonstrate the use of this declarative approach to achieve a well-balanced mix. A knowledge base is compiled with rules and constraints extracted from the literature about what professional music producers and audio engineers suggest creating a good mix. More specially, this work can deliver either a mixed audio file (mixdown) as well as a mixing plan in (human-readable) text format, to serve as a starting point for producers and audio engineers to apply this methodology into their productions. Finally this article presents a decibel (dB) and a panning scale to explain how the mixes are generated.

1. INTRODUCTION

The stereo mixing process in a glance, results by placing each independent recorded instrument between two speakers or channels using a three dimensional space in a balanced way; this means, working with depth (volume), width (panorama) and height (frequency). Audio engineers and music producers with specific knowledge and skills usually do this task by using certain software called Digital Audio Workstation (DAW) with no feedback from the computer or any intelligent system. Nevertheless, there are many rules regarding mixing. Some rules describe conventions about volume settings, placing instruments in the panorama or even equalize instruments within certain bandwidth frequencies. This leads to the idea of gathering all these rules and let a computer produce mixes automatically or at least set a starting points that can be modified later in a DAW.

On the other hand, besides the DAWs available to create your mix, related work has already explored several tasks that compel the mixing procedure in an automatic way such as: Levels balancing, automatic panning, dynamic range compression and equalization, among others [1–6].

From this point, there are some automatic mixing systems that integrate the tasks previously mentioned by using one of three major areas of study [7] which are:

- Machine Learning techniques, trained on initial mixes to apply the learned features on new content.
- Grounded Theory approach, aims to acquire basic mixing knowledge including psychoacoustic studies and user preferences to eventually transfer this compilation to an intelligent system, and
- Knowledge Engineering, feeds an intelligent system with already known rules and constraints.

All these related work uses low-level features (audio analysis) and their mixing decisions are solely based on this information. So far none has used a declarative proposal as an alternative to mix multiple tracks. As an alternative, this paper shows an innovative way to create mixes using Answer Set Programming (ASP) [8] as part of the Knowledge Engineering category. This proposal uses high-level information extracted from the literature including audio engineering books, websites and peer-reviewed papers such as [7, 9–14] without taking in consideration (yet) low-level information.

Previous work has shown the use of a declarative language such as ASP to compose diverse types of musical styles [15, 16] including lyrics that matches the composition [17]. Other works proposes to build and compose chords progressions and cadences [18, 19], create score variations [20] or fill the spaces to complete a composition [21]. The reason why ASP fits perfectly for this type of problem is because of its declarative approach that allows in a compact way, describing the problem rather than the form of getting the solution. Another benefit is the possibility to add or change rules without taking care of their order of execution. Also mixing is a high combinatorial problem meaning that there are many ways to do a mix [9] independently of the audio files used. Using a generate-and-test approach may deliver valid, and even not yet proposed (heard) results.

The goal of this work is to demonstrate the use of rules, the easy modelling for mixing rules and engineering knowledge representation, as well as the high-performance of ASP to create a balanced studio mix and a human-readable plan file. This audio mix is not intended to be closed to a specific genre, although only a fixed set of instruments is used. Audio input files can be given to generate the mixdown file and render a WAV file using Csound [22]. Besides, due the lack of real-life examples settings or common practices describing professional's approaches on how to balance the audio files [7], this work proposes a decibel (dB) and a panorama scale to place each instrument in the sound field. To prove the basic modeling of a mixing procedure, the scope of this work focuses on balance and panning only, leaving further processors such as dynamic range compression, equalization and filtering for future developments.

The result of this work is a static mixdown file, which is a rendered file with at most 16 instruments mixed. The user can adjust the number of instruments in the mix. According to Gibson [10], the most common instruments in a mix are: bass drum (also known as kick), snare, hi hats, hi and low toms, left and right overs, ride cymbal, a bass line, lead and rhythmic guitar, lead and back vocals, piano, violin and cello. These fix but vast number of instruments is sufficient enough to illustrate the goal of this paper.

This document starts by giving a brief explanation of the concepts that are involved the mixing process. The next section describes what ASP is and how the mix is modeled in a logic form. Finally, this paper concludes with an evaluation and results of the mix and the plan files as well as a discussion on further work.

2. THE MIXING PROCESS

The mixing process involves many expert tasks that accurately described, can be implemented in software or hardware [23]. All this expert knowledge extracted from the literature consists of sets of rules, constraints, conventions and guidelines to recreate a well-balanced mixdown. As stated before the mixing process can be divided in three major areas: Dynamics (depth), Panning (Width) and Frequencies (Height) and it is common to find in the literature [7, 9–14] that this sequence persists as the main flow of the mix. The first step in a mix is to handle the depth of the track.

2.1 The Dynamics inside the Mix

The dynamics represents the emotion that the mix creates and it is made by adjusting the volumes of different tracks respecting which instruments should be louder than others. The first step that most experts state is the need to have a lead instrument which will sound the loudest compared to the remaining instruments in the mix. The reason behind this is that every song has its own personality and is mixed based on the lead instrument. Once the lead instrument is selected, the next step will define the volumes of the remaining instruments.

David Gibson states [10] that a good practice is to set each instrument into one of six apparent volume levels, but before setting the amount of volume of each instrument, first let's introduce about sound pressure level, decibels, and amplitude ratio. When you raise a fader on a mixing board, you are raising the voltage of the signal being sent to the amp, which sends more power to the speakers, which increases the Sound Pressure Level (SPL) in the air that your ears hear. Therefore, when you turn up a fader, of course, the sound does get louder [10]. The decibel (dB) is a logarithmic measurement of sound and is the measure for audio signal or sound levels. The relationship between dB and the percentage of the sound states that 0 dB equals a 100% of the recorded (or incoming) sound, in other words the incoming signal remains untouched. If the fader is moved 6 dB up it will double the incoming sound (+6dB = ∼200%) and the same occurs in the opposite way, if the fader is decreased 6 dB below 0 dB it will represent only half power of the original sound (-6dB = ∼50%). This comes from the relation between the Amplitude Ratio-Decibel function which is $dB = 20\log_{10} N$, where dB is the resultant number of decibels, and N equals the Amplitude Ratio. This formula states that an amplitude ratio of 1, equals 0 dB, which is the maximum volume of an incoming signal without distortion. The Table 1 demonstrates that 41 dB represents a full-scale domain from the incoming sound percentage (ISP) from 1 to 100%.

Table 1. Amplitude Ratio-Decibel Conversion Table

dB	ISP (∼%)	Amplitude Ratio
+6	200	1.9953 (∼2)
0	100	1
-1	89	0.891
-2	79	0.794
-3	71	0.708
-6	50	0.501 (∼1/2)
-10	32	0.316
-20	10	0.1
-30	3	0.0316
-40	1	0.01

The six apparent volume levels that Gibson proposes are conventions to set the recorded instruments into the sound field. Table 2 shows a scale for this six volume levels used in this work. According to Gibson, the room contains six levels; nevertheless the literature doesn't count a real dB scale to map this room. Most of the literature only display analogies about how far or how close an instrument should be from another. The scale in Table 2 maps the room using the information in Table 1 as reference. The half of the room (between levels 3 and 4) equals the half of the sound which is -6 dB (50%) and from this point the rest of the values are divided in a uniform way to cover the six levels. After balancing all the instruments by placing a dB or SPL value according to the previous information, the next step is panning.

Table 2. Six Volume Levels- dBs Range Table

Volume Level	dBs (From)	dBs (To)	ISPs
1	0	-1.83	100 - 81
2	-1.94	-4.29	80 - 61
3	-4.44	-7.74	60 - 41
4	-7.96	-19.17	40 - 11
5	-20.0	-24.44	10 - 6
6	-26.02	-40.0	5 - 1

2.2 How Do We Turn The Pan Pot?

Moving the instruments between the speakers from left to right does the panning or the positioning of each instrument in the sound field (normally with the use of a physical or digital pan pot) with the purpose to gain more clarity in the mix. The panning is not the result of treating each channel individually, instead, it is the result of the interaction between those channels.

The engineer's job is to maintain the balance of the track across the stereo field and one recommended technique is setting the sounds apart with similar frequency ranges. Another strong suggestion is to avoid hard panning which is panning an instrument fully right or left. Low-frequency tracks must not be panned out of the center and the reason is to ensure that the power of the low-frequency tracks are equally distributed across the speakers. That's why the lowest frequency instruments are not panned like the bass line and the kick.

This work uses the -3 dB, equal power, sine/cosine panning law, which consist of assigning values from -1 (hard left panning) to +1 (hard right panning) to determine the relative gain of the track between channels. This law is the most common to use according to audio engineering literature [11]. The formulas to calculate the amount of energy that each speaker will generate in both, left and right channels are:

$$LeftGain = cos(\pi(p+1)/4) \tag{1}$$

$$RightGain = sin(\pi(p+1)/4) \tag{2}$$

To get the value of "p" which corresponds to the panning value of a single instrument, a scale which contains 21 possible values is proposed, 10 assigned to the left (from -1 to -0.1), 10 to the right (from 0.1 to 1) and 1 value for centered instruments (0 for center panning).

3. ANSWER SET PROGRAMMING

ASP is a logic-programming paradigm on the use of non-monotonic reasoning, oriented on solving complex problems originally designed for the Knowledge Representation and Reasoning domain (KRR). ASP has become very popular in areas, which involve problems of combinatorial search using a considerable amount of information to process like Automated Planning, Robotics, Linguistics, Biology and even Music [24]. ASP is based on a simple yet expressive rule language that allows to easily model problems in a compact form. The solutions to such problem are known as *answer sets* or stable models [25], nowadays handled by high performance and efficient Answer Set Solvers [26].

This paper gives a small introduction to the syntax and semantics of logic programming and ASP including the normal or basic *rule* type, *cardinality rules* and *integrity constraints* . For further reading and a more in-depth coverage of the expressions shown in this section, refer to [8,24,27]. A normal *rule* expression r is in the form:

$$a_0 \leftarrow a_1, \ldots, a_m, \sim a_{m+1}, \ldots, \sim a_n \tag{3}$$

where a_i, for $0 \leq m \leq n$, is an *atom* of the form $p(t_1, \ldots, t_k)$ with predicate symbol p, and $t_1, \ldots, t_k$ are terms such as constants, variables, or functions. The $\sim a_i$ stands for default negation, meaning that a literal *not A* is assumed to hold unless atom A is derived to be true. Letting the expression $head(r) = a_0$, $body(r)^+ = \{a_1, \ldots, a_m\}$, and $body(r)^- = \{a_{m+1}, \ldots, a_n\}$, r denoted by $head(r) \leftarrow body(r)^+ \cup \{\sim a \mid a \in body(r)^-\}$. A *logic program* P consist in a set of rules of the form (3). The *ground instance* (or *grounding*) of P, denoted by $grd(P)$, is created by substituting all variables in a with some elements inside the domain of P. This results in a set of all ground rules constructible from rules $r \in P$. A satisfiable ground rule r of the form (3) contains a set X of ground atoms if $body(r)^+ \subseteq X$ and $body(r)^- \cap X = \emptyset$ refering that $a_0 \in X$. If X satisfies every rule $r \in grd(R)$, X is a *model* of P and X is an *answer set* of P if X is a subset-minimal model of $\{head(r) \leftarrow body(r)^+ \mid r \in grd(P), body(r)^- \cap X = \emptyset\}$.

For *cardinality rules* and *integrity constraints* the expressions are in the form

$$h \leftarrow a_1, \ldots, a_m, \sim a_{m+1}, \ldots, \sim a_n \tag{4}$$

where a_i, for $1 \leq m \leq n$, an *atom* of the form $p(t_1, \ldots, t_k)$ with predicate symbol p, and $t_1, \ldots, t_k$ are terms such as constants, variables, or functions. The head h can be either $\perp$ for *integrity constraints* or of the form $l\,\{h_1, \ldots, h_k\}\,u$ for *cardinality constraint* in which l, u are integers and $h_1, \ldots, h_k$ are atoms. Similarly a ground cardinality rule is satisfied by a set X of ground atoms if $\{a_1, \ldots, a_m\} \subseteq X$ and $\{a_{m+1}, \ldots, a_n\} \cap X = \emptyset$ implying $h = l\,\{h_1, \ldots, h_k\}\,u$ and $l \leq |\{h_1, \ldots, h_k\} \cap X| \leq u$; finally a ground integrity constraint if $\{a_1, \ldots, a_m\} \not\subseteq X$ or $\{a_{m+1}, \ldots, a_n\} \cap X \neq \emptyset$. Both ground cardinality (in a rule's head) and ground integrity constraints are satisfied under certain conditions like only satisfied by a set of ground atoms X if X does not satisfy its body [1] . In case of a ground cardinality constraint if the statement before is not fulfilled, X must contain at least l and at most u atoms of $\{h_1, \ldots, h_k\}$. For example, given the following program in ASP code:

```
a :- b, c.
b :- not d.
c.
e :- f.
```

Listing 1. ASP Basic Code

the unique answer set is `{a, b, c}`. The default-negated value `not d` satisfies `b`, because there is no evidence of `d` as a fact. Saying this, both facts `c, b` satisfies the first rule to get `a`. Note that `f` cannot satisfy `e` because there is no prove of `f` to be true.

The following section shows the ASP programs (encodings) that lead to possible configurations of a mix which may include rules with arithmetical functions i.e., {'+', '−', '⋆', '/'} and also comparison predicates {'=', '!=','<', '<=','>=','>'}. This work uses the state-of-the-art grounder *gringo* [28] and solver *clasp* [26].

4. MIXING WITH ASP

The procedure about how to achieve a mixed audio file or a mixdown file is divided in 6 parts being the instruments

[1] For cardinality constraints, it does not hold that both $\{a_1, \ldots, a_m\} \subseteq X$ and $\{a_{m+1}, \ldots, a_n\} \cap X = \emptyset$

in the mix, the lead instrument, the room definition, setting dynamics, panning instruments and stereo mix evenness.

The instruments inside the mix can be user defined and as stated before there are 16 possible instruments to mix. The resultant mix can contain any number of instruments from the mentioned list. Each desirable instrument to appear in the mix must be reflected as a fact `trackOn(I)`, where `I` is the instrument name like “kick”, “snare”, “piano” or “bass”. Listing 2 shows part of the instruments definition.

```
trackOn(kick).
trackOn(snare).
trackOn(hihats).
trackOn(bass).
trackOn(piano).
trackOn(leadVocals).
trackOn(leadGuitar).
```

Listing 2. Instruments Definition

Similarly to the previous part, the lead instrument can be defined as a fact as `leadInstrument(I)` or a rule can deduct the lead instrument (if not defined) as shown in line 1 of Listing 3. From lines 3 to 12 a set of integrity constraints is coded in order to forbid certain instruments to take a lead position. In other words, from the 16 instruments defined, only six are candidates to lead the mix (piano, lead guitar, lead vocals, bass, cello and violin).

```
1   1 { leadInstr(I) : trackOn(I) } 1.

3   :- leadInstr(kick).
4   :- leadInstr(snare).
5   :- leadInstr(hihats).
6   :- leadInstr(hitoms).
7   :- leadInstr(lowtom).
8   :- leadInstr(leftOvers).
9   :- leadInstr(rightOvers).
10  :- leadInstr(ride).
11  :- leadInstr(rhythmGuitar).
12  :- leadInstr(backVocals).
```

Listing 3. Lead Instrument and Integrity Constraints

The third part of the encoding consists of the room definitions as stated in Table 2. The encoding below contains the range of dB for each volume level.

```
volLeveldB(1,  -1..0).
volLeveldB(2,  -3..-2).
volLeveldB(3,  -6..-4).
volLeveldB(4, -10..-7).
volLeveldB(5, -15..-11).
volLeveldB(6, -20..-16).
```

Listing 4. Room Definition in ASP

The following encoding (Listing 5) describes in a generate-and-test approach, how each instrument gets a volume level and a dB value (lines 3-4). This section also contains an extract set of constraints that according to experts, some instruments must sound louder than others meaning that a certain instrument must respect certain volume levels only. The lines 10-12 states that there must not be other instrument louder than the lead instrument, placing the lead sound in the volume level 1. The lines 14-16 says if an instrument `I` is not lead, it cannot be at level 1. Other type of instruments like the hi hats can play around levels 2 and 5 (line 20), contrary to the kick drum that only can be placed at levels 2 and 3 (line 18). The integrity constraints from lines 24-31 also limit these instruments to a certain volume level. The main difference is that this constraint only applies if the instrument is not lead.

```
1   volumeLevel(1..6).

3   1 { instrVolumeLevel(I,VL) :
4       volumeLevel(VL) } 1 :- trackOn(I).

6   1 { instrDB(I,DB) :
7       volumeLeveldB(VL,DB) } 1 :-
8           instrVolumeLevel(I,VL).

10  :- instrVolumeLevel(LI,VL),
11      trackOn(LI),
12          leadInstr(LI), VL != 1.

14  :- instrVolumeLevel(I,VL),
15      trackOn(I),
16      not leadInstr(I), VL == 1.

18  :- instrVolumeLevel(kick,VL), VL>3.
19  :- instrVolumeLevel(snare,VL), VL>3.
20  :- instrVolumeLevel(hihats,VL), VL>5.
21  :- instrVolumeLevel(hiToms,VL), VL<3.
22  :- instrVolumeLevel(backVocals,VL),VL<2.

24  :- instrVolumeLevel(bass,VL), VL>3,
25      not leadInstr(bass).
26  :- instrVolumeLevel(leadGuitar,VL),VL>3,
27      not leadInstr(leadGuitar).
28  :- instrVolumeLevel(leadVocals,VL),VL>3,
29      not leadInstr(leadVocals).
30  :- instrVolumeLevel(piano,VL),VL>4,
31      not leadInstr(piano).
```

Listing 5. Dynamics

```
panDirection(left;right;center).

maxPanValue(10).
panLevels(1..MPV) :- maxPanValue(MPV).

1 { instrPanDirection(I,PD) :
    panDirection(PD) } 1 :-
        trackOn(I).

:- instrPanDirection(kick,PD),
    PD != center.
:- instrPanDirection(bass,PD),
    PD != center.
:- instrPanDirection(leadVocals,PD),
    PD != center.
:- instrPanDirection(snare,PD),
     PD != center.

:- instrPanDirection(hihats,center).
:- instrPanDirection(lowTom,center).
:- instrPanDirection(leadGuitar,center).
:- instrPanDirection(rhythmGuitar,center).
:- instrPanDirection(piano,center).
:- instrPanDirection(violin,center).
:- instrPanDirection(rightOvers,center).
```

Listing 6. Pan Directions

The panorama (in a similar way to the volume level decisions) starts generating and tests the possible outcomes by picking one of three possible direction of the stereo field, left, center or right (lines 1-8). Additionally each instrument in the mix needs to respect the panning constraints. Listing 6 shows an example of these constraints, stating that the kick, bass, lead vocals and the snare instruments must respect a centered position (lines 10-17). On the other side, there are instruments that cannot be played in the center (lines 19-25). Listing 7 also contains another set of integrity constraints to respect sides. Examples of these are: The left overs cannot be on the right and vice versa (lines 1-4). If the two guitars, lead and rhythm are in the mix, without evidence that the piano is the mix, the guitars must be on different sides (lines 6-10). Also if there is no evidence that the lead guitar exists, but both rhythm guitar and piano are facts, pan both in different sides too (lines 12-16).

```
 1  :- instrPanDirection(leftOvers,  PD),
 2      panDirection(PD), PD != left.
 3  :- instrPanDirection(rightOvers, PD),
 4      panDirection(PD), PD != right.

 6  :- instrPanDirection(rhythmGuitar,PD),
 7      instrPanDirection(leadGuitar, PD),
 8      not trackOn(piano),
 9      trackOn(leadGuitar),
10      trackOn(rhythmGuitar).

12  :- instrPanDirection(rhythmGuitar,PD),
13      instrPanDirection(piano,PD),
14      not trackOn(leadGuitar),
15      trackOn(rhythmGuitar),
16      trackOn(piano).
```

Listing 7. Side Panning Constraints

After picking sides, the next step is to figure out how far left or right each instrument will be panned (Listing 8). The firsts four lines states that if the instrument is not centered, choose one pan level and place it there. The literature does not force to pan an instrument to a specific position, except from the ones in the center. This open opportunities to play around and listen some instruments in different positions on each mixing configuration. To avoid hard panning, lines 6 to 9 are in charge to satisfy this constraint. Similarly, lines 11 to 14 avoids that two instruments both panned left or right, stay at the same panning level.

```
 1  1 { instrumentPanningLevel(I,PD,PL)
 2      : panLevels(PL) } 1  :-
 3      instrumentPanDirection(I,PD),
 4      PD != center.

 6  :- instrPanningLevel(I,right,MPV),
 7      maxPanValue(MPV).
 8  :- instrPanningLevel(I,left ,MPV),
 9      maxPanValue(MPV).

11  :- instrPanningLevel(I1,PD,PL),
12      instrPanningLevel(I2,PD,PL),
13      I1 != I2,
14      PD != center.
```

Listing 8. How Far Left or Right Constraints

Finally, the part six counts the total number of instruments, the ones in the center, left and right (Listing 9). The last four lines (19-22) avoid that the mix is uneven or unbalanced between the instruments positioned in the left and right. The mix must not be charged to one side.

```
 1  countTracksOn(X)   :-
 2      X = #count{0,trackOn(I):trackOn(I)}.

 4  countPanCenter(X) :-
 5      X = #count{0,
 6          instrPanningLevel(I,center,PL) :
 7          instrPanningLevel(I,center,PL)}.

 9  countPanLeft(X)   :-
10      X = #count{0,
11          instrPanningLevel(I,left,PL) :
12          instrPanningLevel(I,left,PL)}.

14  countPanRight(X)  :-
15      X = #count{0,
16          instrPanningLevel(I,right,PL) :
17          instrPanningLevel(I,right,PL)}.

19  :- countPanLeft(N) ,
20      countTracksOn(X), N >= X/2.
21  :- countPanRight(N),
22      countTracksOn(X), N >= X/2.
```

Listing 9. Making the Mix Even

Given this knowledge base it is possible to get a configuration of a balanced mix using volumes and pan only. The solution output from ASP (depending on the number of instruments) can contain facts such as: `countTracksOn(8) countPanCenter(3) countPanLeft(2) countPanRight(3) leadInstrument(leadGuitar) intrumentDB(hihats,-2) intrumentDB(bass,-2) instrumentPanningLevel(kick,center,0) instrumentPanningLevel(rhytmicGuitar,left,7) instrumentPanningLevel(lowTom,right,1) instrumentPanningLevel(leadGuitar,right,3)`.

The facts above are an extract of an answer set. This notation can be parsed into human-readable text for the mixdown plan and to Csound code to render the WAV file.

5. TESTING AND RESULTS

To test the performance towards an automated mixing tool, five songs from different genres were mixed using only the rules mentioned. Rendered mix files were generated starting from the grounding and solving processes by gringo and clasp respectively. Followed by two Perl file parsers, one to convert the answer set to a human-readable plan and the other to parse the output to Csound code. There is no audio treatment in Csound, just indicating the volume in dB and the resultant panning values. A Csound function automatically determines if the incoming file is mono or stereo in order to use that file in the mix. No low-level features are extracted and no further processors are applied to the mixes.

The audio stems used for testing are high quality recorded publicly available from the Free Multitrack Download Library by Cambridge Music Technology website [2].

The mixes lasts between 20-60 seconds and the number

[2] *http://cambridge-mt.com/ms-mtk.htm*

of tracks varies depending on the genre. For each song, four different answers were asked to the solver in order to test that:

1. All the mixdown files generated didn't cause clip (distortion). Even though it is possible to tell Csound to leave certain headroom below 0 dB (also suggested by the literature).
2. All the selected instruments were heard in the mix.
3. In fact all the mixes respect a balanced distribution of the instruments through the sound space.
4. No inconsistencies were found when running and switching between different types of instruments. In other words, there are no rules that contradict others.
5. Using the aforementioned rules, this work can deliver a vast number of different and valid mixing configurations.

Less than 100 ASP code lines (not included comments and white spaces) are enough to develop a rule-based approach for multitrack stereo mixing. Also letting a non-deterministic solver to propose thousands possible answers in short amount of time (Over 1000 different results in less than 0.031 seconds) gives the option to ask for another mixing plan without changing the input audio files. This value was obtained by running Clingo 5 on a Intel based machine with 3.07GHz CPU with 15 GB RAM.

One of the main features discovered is that taking different stems with some files nearly recorded at 0 dBFS, the rendered file doesn't clip, even as it is mentioned, this work doesn't read audio spectral information.

The results shows that the track stems worked better if they were (peak) normalized or recorded nearly 0 dBFS. Examples of input tracks with a highest peak of -5 dB, got lost inside the mix and depending on their loudness some stems couldnt́ be heard. Examples of barely audible or none audible are the kick, snare, toms and particularly the kick fights against the bass in the low-end. Masking issues were generated because of the lack of EQ, but certain clarity was achieved because of the panning rules. This escenario can be dealt by having two options, being the first one to use normalized tracks only or analyze data such as peak level, crest factor, loudness and root mean square (RMS) level from a track. Adding this new knowledge will treat each instrument in a different way.

Changing the lead instrument between executions showed significant differences in the mixes. Not having the lead vocals in the (instrumental) mixes allowed the guitars and piano play around different positions compared with the mixes that use vocals. Also when vocals are played in the mix different results showed clarity in instruments more closed to fully right or left.

Also no inconsistencies were found during all the executions. This knowledge base can produce mixes up to 16 instruments and be a starting point to add more instruments.

The generated mixes with their mixing plans can be found in *http://soundcloud.com/flavioeverardo-research/sets*.

6. DISCUSSION AND FURTHER WORK

This work introduced a Knowledge-Engineered approach with the use of ASP for proposing mixes in an automatic way using high-level information. This work can be the upper layer from one of the existing work which rely on low-level features, convinced that this integration will lead to more real-life mixes. On the other hand this work can be extended to a formal system and it is necessary to integrate low-level features such as cross-adaptive methods [2, 29] and even sub grouping mixing rules [30].

Saying this, another imminent goal is the development of the third axis for frequencies equalization including filters like bandpass, low-pass, hi-pass or shelf to solve inter-channel masking. Also the addition of a processing chain that includes compressor, and time-based effects like delay and reverb.

For further code testing, several configurations can be developed to fulfill different mixing requirements for specific music styles. An example of this is giving the openness to define different "room" types (different volume level and dB scale) and panning values depending on the music genre. So far this work uses rules for most common instruments and the addition of new instruments like digital synthetizers and other electronic-music related instruments like pads, leads and plucks (to mention some of them) are necessary to open this tool to other music styles and mixes.

The use of ASP leaves the option to change and modify rules and constraints to describe other parts of the mix. Also this allows keeping the knowledge separated from the sound engine. This knowledge base can be used independent from Csound allowing other options for the sound treatment. Adding to this, one further development is that different filter types, compressors and other audio effects can be coded in Csound. Afterwards add this information as rules so different instruments can be treated with different audio processing.

Lastly but not least, the benchmarking and assessment process against other automated mixing systems, professional audio engineers and subject test rating (like shown in [11]) needs to be done in order to evaluate the efficiency of the rules and constraints explained in this work.

7. REFERENCES

[1] E. P. Gonzalez and J. Reiss, "Automatic mixing: live downmixing stereo panner," in *Proceedings of the 7th International Conference on Digital Audio Effects (DAFx'07)*, 2007, pp. 63–68.

[2] E. Perez-Gonzalez and J. Reiss, "Automatic equalization of multichannel audio using cross-adaptive methods," in *Audio Engineering Society Convention 127*. Audio Engineering Society, 2009.

[3] ——, "Automatic gain and fader control for live mixing," in *Applications of Signal Processing to Audio and Acoustics, 2009. WASPAA'09. IEEE Workshop on*. IEEE, 2009, pp. 1–4.

[4] S. Mansbridge, S. Finn, and J. D. Reiss, "An autonomous system for multitrack stereo pan position-

ing," in *Audio Engineering Society Convention 133*. Audio Engineering Society, 2012.

[5] ——, "Implementation and evaluation of autonomous multi-track fader control," in *Audio Engineering Society Convention 132*. Audio Engineering Society, 2012.

[6] J. J. Scott and Y. E. Kim, "Instrument identification informed multi-track mixing." in *ISMIR*. Citeseer, 2013, pp. 305–310.

[7] B. D. Man and J. D. Reiss, "A semantic approach to autonomous mixing," in *APR13, 2013. Ma, et al. PARTIAL LOUDNESS IN MULTITRACK MIXING*.

[8] C. Baral, *Knowledge representation, reasoning and declarative problem solving*. Cambridge university press, 2003.

[9] B. Owsinski, *The mixing engineer's handbook*. Nelson Education, 2013.

[10] D. Gibson, "The art of mixing: A visual guide to recording," *Engineering, and Production*, vol. 236, 1997.

[11] B. De Man and J. D. Reiss, "A knowledge-engineered autonomous mixing system," in *Audio Engineering Society Convention 135*. Audio Engineering Society, 2013.

[12] A. U. Case, *Mix smart*. Focal Press, 2011.

[13] M. Senior, *Mixing secrets for the small studio*. Taylor & Francis, 2011.

[14] E. Perez_Gonzalez and J. Reiss, "A real-time semi-autonomous audio panning system for music mixing," *EURASIP Journal on Advances in Signal Processing*, vol. 2010, no. 1, p. 436895, 2010.

[15] G. Boenn, M. Brain, M. De Vos, and J. Ffitch, "Automatic music composition using answer set programming," *Theory and practice of logic programming*, vol. 11, no. 2-3, pp. 397–427, 2011.

[16] E. Pérez, F. Omar, and F. A. Aguilera Ramírez, "Armin: Automatic trance music composition using answer set programming," *Fundamenta Informaticae*, vol. 113, no. 1, pp. 79–96, 2011.

[17] J. M. Toivanen *et al.*, "Methods and models in linguistic and musical computational creativity," 2016.

[18] S. Opolka, P. Obermeier, and T. Schaub, "Automatic genre-dependent composition using answer set programming," in *Proceedings of the 21st International Symposium on Electronic Art*, 2015.

[19] M. Eppe16, R. Confalonieri, E. Maclean, M. Kaliakatsos, E. Cambouropoulos, M. Schorlemmer, M. Codescu, and K.-U. Kühnberger, "Computational invention of cadences and chord progressions by conceptual chord-blending," 2015.

[20] F. O. E. Pérez, "A logical approach for melodic variations." in *LA-NMR*, 2011, pp. 141–150.

[21] R. Martín-Prieto, "Herramienta para armonización musical mediante Answer Set Programming," http://www.dc.fi.udc.es/~cabalar/haspie.pdf+, University of Corunna, Spain, Tech. Rep., 02 2016.

[22] R. Boulanger *et al.*, "The csound book," 2000.

[23] J. D. Reiss, "Intelligent systems for mixing multichannel audio," in *Digital Signal Processing (DSP), 2011 17th International Conference on*. IEEE, 2011, pp. 1–6.

[24] M. Gelfond and Y. Kahl, *Knowledge representation, reasoning, and the design of intelligent agents: The answer-set programming approach*. Cambridge University Press, 2014.

[25] P. Simons, I. Niemelä, and T. Soininen, "Extending and implementing the stable model semantics," *Artificial Intelligence*, vol. 138, no. 1-2, pp. 181–234, 2002.

[26] M. Gebser, B. Kaufmann, A. Neumann, and T. Schaub, "clasp: A conflict-driven answer set solver," in *International Conference on Logic Programming and Nonmonotonic Reasoning*. Springer, 2007, pp. 260–265.

[27] V. Lifschitz, "What is answer set programming?." in *AAAI*, vol. 8, no. 2008, 2008, pp. 1594–1597.

[28] M. Gebser, R. Kaminski, A. König, and T. Schaub, "Advances in *gringo* series 3," 2011, pp. 345–351.

[29] J. Scott, "Automated multi-track mixing and analysis of instrument mixtures," in *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014, pp. 651–654.

[30] D. M. Ronan, H. Gunes, and J. D. Reiss, "Analysis of the subgrouping practices of professional mix engineers," in *Audio Engineering Society Convention 142*. Audio Engineering Society, 2017.

Explorations in Digital Control of MIDI-enabled Pipe Organs

Johnty Wang
Input Devices and Music
Interaction Laboratory/CIRMMT
McGill University
johnty.wang@mail.mcgill.ca

Robert Pritchard
School of Music
University of British Columbia
bob@mail.ubc.ca

Bryn Nixon
Ryerson United Church
Vancouver, Canada
brynnixon@icloud.com

Marcelo M. Wanderley
Input Devices and Music
Interaction Laboratory/CIRMMT
McGill University
marcelo.wanderley@mcgill.ca

ABSTRACT

This paper describes the use of a MIDI-enabled pipe organ console at Ryerson United Church in Vancouver for music service during worship, as well as a custom built dedicated librarian and performance software that opens up possibilities for exploration in alternative control of the instrument via various gestural interfaces. The latter provides new possibilities for expression and extended performance practice including dance and interactive installations. Future work on both the artistic use of the system as well as technical development of the interfacing system is presented.

1. INTRODUCTION

The pipe organ, unlike most other acoustic instruments, allows modification of its *mapping* [1]. In other words, it is possible to change the behaviour of the instrument through stops and couplers, affecting the sound produced when the keyboard manuals and pedals are played. Most modern organs are controlled with electronic systems that provide faster and more scalable performance [2]. However, in addition to technical improvements, digital systems also allow easy addition of external interfaces such as MIDI, which support extended control techniques using other devices. A MIDI enabled pipe organ's console can be used as an interface that allows performer input to be recorded or transmitted to other sound generators, and the sound producing mechanism of the instrument effectively becomes a synthesizer that can be controlled with compatible signals. In this paper, we describe the use of a MIDI-enabled pipe organ console for use in worship service as well as explorations in alternative control.

2. RELATED WORK

The Yamaha Disklavier [1] is an example of a MIDI-enabled instrument that can be controlled externally as a synthesizer. In the case of the pipe organ, the spatial arrangement of the pipes and wide spectral range provide additional possibilities over the single array of strings of the piano.

The Organ Augmented Reality project [3] extends the output of a pipe organ through real-time projection mapping of audio visualizations as well as sound processing and spatialization.

Chris Vik's "Carpe Zythum" [4] is a composition that makes use of a Microsoft Kinect sensor to provide gestural control on a similar organ console. The work described in this paper, in contrast, makes use of the MIDI functionality for musical service at the church, as well as the construction of a platform to support new performances.

3. RYERSON UNITED CHURCH ORGAN

Ryerson United Church, in addition to being a place of worship, hosts a large number of musical events including the "Sundays at 3" concert series, the Pnuema Children's Choir, Dunbar Ryerson Voices Choir (a large community choir that performs 2 oratorios each year with orchestra), and a number of concerts by various professional and amateur choirs in the community.

The organ is, shown in Figure 1 is 4 manual instrument built by the Canadian builder Casavant Frères in 1964. The modern control system is installed by Steve Miller of Solid State Organ Systems [2]. In 2005 the organ was rebuilt and part of the process included the addition of a MIDI-capable controller. In 2012 the actual MIDI interface was physically exposed, providing an externally accessible interface of the control system. The built-in features of the current console include a sequencer that can store and playback recordings on a USB stick, and extended capabilities include connection with a tablet application [3] that can be

[1] http://www.disklavier.com/
[2] http://www.ssosystems.com/
[3] http://www.ssosystems.com/pages/palette.html

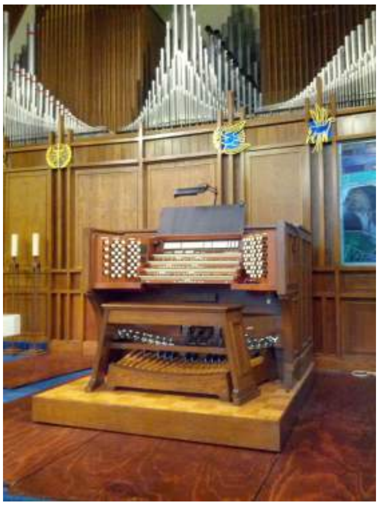


Figure 1. The Organ at Ryerson United Church.

used to control various settings of the instrument. The MIDI output port of the console will emit messages pertaining to keyboard and pedal action using note messages, as well as system changes representing the state of the couplers and stops via System Exclusive (SysEx) commands. The MIDI input provides reciprocal features of the output and allows note and system state control from external sources using the same protocol.

4. USE OF MIDI CONSOLE IN MUSIC SERVICE

In this section, we document some of the ways that the MIDI system has been employed by performers in the music service as well as concert and rehearsal settings.

4.1 Rehearsal/Audition Aid

The basic use of the MIDI system with a sequencer has proven to be an invaluable learning tool for all of the organists who practice and perform at Ryerson, since it allows them to record various ways of performing a piece and then play those back in order to hear their registrations, tempi, phrasing etc. as it sounds from the audience's perspective.

4.2 Playback

Since the playback of a MIDI sequence would result in the exact sound produced by a live performance, it is possible to playback recordings to provide background music in the church *as if* it was performed live, far more realistic than any audio reproduction technique. While this is by no means intended to substitute live performance, this feature greatly increases the versatility of the church space.

4.3 Accompaniment

There has been some experimentation with performing Sunday service 'postludes' using the sequencing features of the MIDI, click track and live piano, with performances of Bach, Halley, and Hymn 'preludes'.

A number of choral anthems were performed using the MIDI sequencer, such as the Duruflé Requiem mvts 1 and 2, Bach choral movements from Cantatas. Orchestral parts can be entered into a MIDI sequencer with wind and string parts represented on different divisions or manuals which presents the music as it was written and in a way that cannot be performed by a live player through a transcription of the score.

4.4 Tuning Aid

Finally, due to the large distance between the pipes and the console, the tuning process of a church organ can be quite cumbersome. In the traditional process, a particular stop is activated and either an assistant in the form of a real person (or a device such as a pencil) is used to depress and hold a key on the manual to sound the pipe while another person performs the tuning of the pipe. With this interface, it is possible to use a wireless controller and activate any stop or key remotely, thereby making the process much easier with a single person.

5. EXPLORATIONS IN ALTERNATIVE CONTROL

To provide an entry point in exploring alternative control of the instrument, we first built an interface patch to expose access of the internal configuration of the console. The keyboard/pedal and swell controls are done with standard MIDI note and volume messages which are directly accessible in most programming environments for digital media, but the internal state of the active stops can only be accessed via System Exclusive (SysEx) messages. We have implemented a Max/MSP driver patch to translate between the internal protocol used by the instrument and the user interface. From there, additional user applications performing various mapping between gestural signal input, mapping patches and/or external signals can then communicate locally within *send* and *receive* objects, or via network protocols such as Open Sound Control (OSC) [5].

5.1 Overview of Max/MSP Interface Patch

While the Ryerson organ uses MIDI SysEx protocol and can accept MIDI messages, the low speed of MIDI communication can preclude the use of the MIDI transmission speed on such a complex instrument. Instead, the core transport medium is packet-based Ethernet. The organ console is scanned at approximately 500 Hz and a packet containing the complete state of the organ is sent whenever a change is detected. Thus, the complete state of the organ can be changed with one packet. To accommodate the MIDI protocol, 8k transmission buffers are used. However, with no pacing on the interrupt-driven transmission buffer, care must be taken to avoid flooding the system with data.

The full configuration of the organ's stops and couplers is represented by a 22 byte SysEx message, as shown in Figure 2. The first two bytes contain the header and manufacturer ID. Then 3 bytes are used to select the bank or group of the configuration. Following are 16 bytes of configuration data appended by the single byte corresponding to the terminating SysEx footer.

Each stop is associated with a specific bit in the 16 bytes for a given bank, so activating and deactivating a stop con-

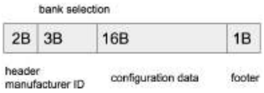


Figure 2. SysEx message format

sists of setting a specific bit to be on or off. When any stop is toggled on or off, its bit value is added to or subtracted from the current status of all stops. This change is immediately read and transmitted as a packet containing the complete state of the organ.

Presets can be created by storing a packet of the organ's current state. When the packet is retrieved and passed to the organ, it automatically updates the state of all stops and combinations. The packet is also passed to the Max/MSP patch which converts the data to bit values and configures the GUI to reflect the organ's state. Any incremental changes can be performed using bitmasking operations so only the stop of interest is affected.

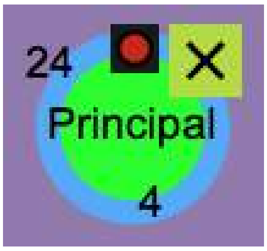


Figure 3. The stop bpatcher

Control of the organ stops was effected through the creation of a Max/MSP organ stop bpatcher, as shown in Figure 3. Each stop bpatcher displays the Casavant Frères stop ID number in the upper left, the name of the stop (Trompette, Bourdon, etc.), and the length of the stop (4, 8, etc.) in the lower centre. Clicking on the blue button toggles the stop on and off, sending a message with the stop's identifying bit turned on or off. This bpatcher also provides visual indication of the status of each stop that may be triggered via the organ console itself.

5.2 Modes of Operation

While the Max/MSP software was designed to work specifically with the Casavant organ at Vancouver's Ryerson United Church, it was coded with an eye/ear towards being useable with similar organs and systems. To that end, the software has learning and performance modes.

In Learning Mode the user arms a particular GUI stop by clicking on its green toggle button. The red LED begins flashing, indicating that the stop is active and ready. The user then pulls the appropriate stop on the organ console. Activating the console stop sends the stop's unique SysEx message to the Max/MSP patch, where the packet is stored in a buffer. The user then clicks on the blue button in the GUI stop, and the packet is saved to a buffer using a *coll* object. The green toggle then turns off, and the red LED stops flashing. Going through this for each stop allows the user to build up the entire stop library, thereby having access to the entire sound capabilities of the instrument.

In Performance mode, the organist plays as per usual, and the status of the organ is reflected in the GUI as the stops are engaged and disengaged. However, external MIDI control is also possible by having a MIDI controller send note-on, note-off, volume, sustain, etc. messages. By sending messages external controllers are also able to turn individual stops on and off, or activate entire combinations of stops.

A screenshot of the main Max/MSP patch is shown in Figure 4. In our explorations thus far we have used OSC endpoints via the *udpreceive* object with manual message routing for external control from user patches and external sensors. This allows us to trigger both note and stop change messages from any device on the network.

5.3 Mapping Examples

We have created a number of preliminary mapping demonstrations with the system[4], which serve to show the potential for building new performances and interactive installations with the system:

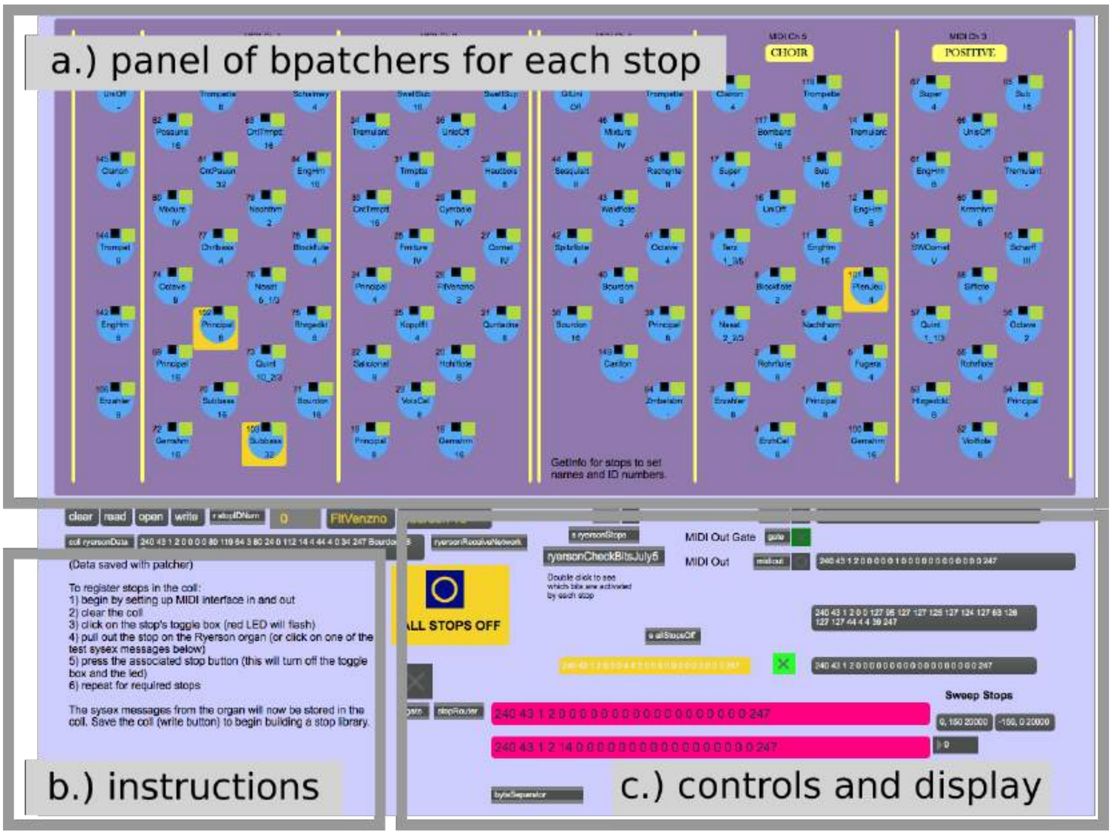
Hand Gesture Control: Using a variety of interfaces such as the Leap Motion[5], a wrist-worn fitness tracker with motion sensors and mobile phones, we have demonstrated preliminary methods of mapping hand motion to note actuations on the instrument. Simple hand-location to note frequency mappings were done, but since sensors provide a wealth of other information (e.g. finger joint position, rates of rotation, drawing gestures on a touch screen, etc.), there are many further possibilities.

Dance: One area of particular interest to us is the ability to use Kinect hardware and custom-written software to track one or two dancers, and use the resulting data to control the organ. The Kinect software, Kinect Controlled Artistic Sensing System (KiCASS) is being developed at the University of British Columbia in capstone projects carried out by undergraduate Engineering students collaborating with the School of Music digital performance ensemble Sonic UBC Laptop Sounds and Sensors[6]. KiCASS runs on a Surface Pro II and can track up to 25 points on each of two dancers. Using a local router it broadcasts the tracking data in OSC format, allowing multiple computers to access the data for use in performance. For dance with the Ryerson organ, a dancer is tracked using the KiCASS system, the resulting OSC data is acquired by a laptop running Max/MSP, and the data is mapped to various performance parameters and sent to the organ through a hardwired MIDI interface. With limb motion, posture, and location the dancer is able to control the generation of MIDI note data and stop configurations.

Prenatal Musical Instrument: A brief experimentation was performed during the development of a prenatal instrument [6] where fetal movements, detected via pressure sensors attached to the belly of the mother, was mapped to trigger notes on the organ.

All of these examples are relatively rudimentary mappings that provide an entry point to more interesting possibilities afforded by the system.

[4] https://www.youtube.com/watch?v=4TuJ5PlTIA0
[5] https://www.leapmotion.com/
[6] https://ubcsubclass2016.wordpress.com



(Data saved with patcher)

To register stops in the coll:
1) begin by setting up MIDI interface in and out
2) clear the coll
3) click on the stop's toggle box (red LED will flash)
4) pull out the stop on the Ryerson organ (or click on one of the test sysex messages below)
5) press the associated stop button (this will turn off the toggle box and the led)
6) repeat for required stops

The sysex messages from the organ will now be stored in the coll. Save the coll (write button) to begin building a stop library.

Figure 4. The Max/MSP Interface Patch showing a.) the panel of bpatchers for each stop, b.)basic usage instructions and c.) miscilleneous controls and status output

5.4 Towards a unified control and mapping framework

The distributed nature of the components of the pipe organ spread across the church building, coupled with the potential for new control input as described above, lends naturally to distributed mapping and control schemes. The Max/MSP patch that was implemented can be seen as a configurable driver application that provides a single point of access to control to the instrument and provides an interface between user input and the lower level SysEx control messages. While it is possible to use end-to-end protocols like OSC to communicate as we have done so far in the examples, explicitly defining connections can quickly become cumbersome in a collaborative setting where multiple sensing devices are spread across multiple machines. A potential platform to work with is the libmapper [7] library, which builds self discovery and dynamic mapping between devices on top of end-to-end protocols like Open Sound Control. Libmapper provides mapping tools that allow simultaneous and potentially different visualization and editing tools of the mapping between devices on the network.

6. FUTURE WORK

For the music service at Ryerson, we intend to continue working pieces that can utilize the orchestral playback and accompaniment features provided by the MIDI system including standard organ repertoire, orchestral and chamber music, and choruses.

One current limitation of the sequence based accompaniment is that they are following the fixed tempo of a recording, and synchronization between the conductor and other instrumentalists and the sequencer are done through click tracks. While it is possible to apply tempo changes as implemented by the“tempo map” feature in many sequencer software, once set it cannot adapt to live performance variations. By employing sequencing software that allows adaptable real-time tempo tracking via gestural input such as NoteAbilityPro [8], it would be possible to provide more flexible accompaniment options during live performance. For the click tracks it may be possible to use haptic instead of audio channels such as the VibroPixel system [9], or the SoundBrenner[7] which eliminate the need for musicians to wear headphones during performance.

Continuing work with the existing Kinect system we are currently working on an interactive dance piece that will be incorporated into a church service. Associating visual gestures in the choreography with sounds emitted by an instrument that is familiar to the congregation provides novel and engaging ways to explore themes presented in the scripture.

Thus far, all of the mapping demonstrations using new controllers have been focused on activating notes. As the large number of different pipes of the organ and their mixtures provide an immensely rich tonal palette, the control of registration (combination of different sets of pipes) is

[7] http://www.soundbrenner.com/

a natural feature of any interactive system involving this instrument.

In terms of software development, our long term goal is to provide an easy to use mapping system that allows artists to access the sonic capabilities of the instrument. By creating and exposing a standard protocol for access along with support with mapping libraries such as libmapper facilitates the use of the system by composers, musicians, and digital media artists.

Since many other organs are equipped with similar control systems, it would be possible to implement the same system in more than one location as well, and provide further possibilities such as teleperformance. In these situations, the self learning feature of the driver patch, extended with the ability to define the number and name of the stops, would be very useful to automatically capture the different stop configuration in other Solid State Organ consoles employing the same SysEx format.

7. CONCLUSION

In this paper we have presented the use and initial explorations in alternative control of a MIDI enabled pipe organ in a church. Ways that this additional piece of technology has extended the existing musical service has been described, along with the technical developments and demonstrations of the system using additional gestural sensing technology showing the potential for creating new performances and interactive installations. Finally, we present our plans for the continued use of the system in service as well as new compositions as well as technical developments to scale up the usability of the system. From a cultural perspective, the work documented in this paper presents a method of utilizing technology and music to foster new forms of dialogue between the religious communities where these instruments have been used traditionally and wider society.

Acknowledgments

The authors would like to thank Steve Miller of Solid State Organ Systems for providing the technical knowledge regarding the MIDI interface, and the music department of Ryerson United Church for providing access to the organ. The primary author is supported by a NSERC-FRQNT Industrial Innovation Scholarship with Infusion Systems, and the Centre for Interdisciplinary Research in Music Media and Technology.

8. REFERENCES

[1] A. Hunt, M. M. Wanderley, and M. Paradis, "The importance of parameter mapping in electronic instrument design," *Journal of New Music Research*, vol. 32, no. 4, pp. 429–440, 2003.

[2] D. Baker, *The Organ: A Brief Guide to Its Construction, History, Usage and Music*, ser. Brief Guide to Its Construction, History, Usage and Music. Shire, 1991.

[3] C. Jacquemin, R. Ajaj, S. Le Beux, C. d'Alessandro, M. Noisternig, B. F. Katz, and B. Planes, "Organ augmented reality: Audio-graphical augmentation of a classical instrument," *Int. J. Creat. Interaces Comput. Graph.*, vol. 1, no. 2, pp. 51–66, Jul. 2010.

[4] C. Vik, "Carpe zythum kinect controlled pipe organ," 2012, [Online; accessed 20-February-2017]. [Online]. Available: https://chrisvik.wordpress.com/2012/04/13/carpe-zythum-kinect-controlled-pipe-organ-full-performance/

[5] M. Wright, "Open Sound Control: an enabling technology for musical networking," *Organised Sound*, vol. 10, no. 03, pp. 193–200, 2005.

[6] A. Pon, J. Wang, S. Carpendale, and L. Radford, "Womba: A musical instrument for an unborn child," in *Proceedings of New Interfaces for Musical Expression*, 2015.

[7] J. Malloch, S. Sinclair, and M. M. Wanderley, "Distributed tools for interactive design of heterogeneous signal networks," *Multimedia Tools and Applications*, vol. 74, no. 15, pp. 5683–5707, 2014.

[8] K. Hamel, "Noteability, a comprehensive music notation system," in *Proceedings of the International Computer Music Conference*, 1998, pp. 506–509.

[9] I. Hattwick, I. Franco, and M. M. Wanderley, "The Vibropixels: A Scalable Wireless Tactile Display System," in *Human Computer International Conference*, Vancouver, 2017.

Live Orchestral Piano, a system for real-time orchestral music generation

Léopold Crestel
IRCAM - CNRS UMR 9912
leopold.crestel@ircam.fr

Philippe Esling
IRCAM - CNRS UMR 9912
philippe.esling@ircam.fr

ABSTRACT

This paper introduces the first system performing *automatic orchestration* from a real-time piano input. We cast this problem as a case of *projective orchestration*, where the goal is to learn the underlying regularities existing between piano scores and their orchestrations by well-known composers, in order to later perform this task automatically on novel piano inputs. To that end, we investigate a class of statistical inference models based on the *Restricted Boltzmann Machine* (*RBM*). We introduce an evaluation framework specific to the projective orchestral generation task that provides a quantitative analysis of different models. We also show that the frame-level accuracy currently used by most music prediction and generation system is highly biased towards models that simply repeat their last input. As prediction and creation are two widely different endeavors, we discuss other potential biases in evaluating temporal generative models through prediction tasks and their impact on a creative system. Finally, we provide an implementation of the proposed models called *Live Orchestral Piano* (LOP), which allows for anyone to play the orchestra in real-time by simply playing on a MIDI keyboard. To evaluate the quality of the system, orchestrations generated by the different models we investigated can be found on a companion website [1] .

1. INTRODUCTION

Orchestration is the subtle art of writing musical pieces for the orchestra, by combining the properties of various instruments in order to achieve a particular sonic rendering [1, 2]. As it extensively relies on spectral characteristics, orchestration is often referred to as the art of manipulating instrumental timbres [3]. *Timbre* is defined as the property which allows listeners to distinguish two sounds produced at the same pitch and intensity. Hence, the sonic palette offered by the pitch range and intensities of each instrument is augmented by the wide range of expressive timbres produced through the use of different playing styles. Furthermore, it has been shown that some instrumental mixtures can not be characterized by a simple summation of their spectral components, but can lead to a unique *emerging timbre*, with phenomenon such as the orchestral blend [4]. Given the number of different instruments in a symphonic orchestra, their respective range of expressiveness (timbre, pitch and intensity), and the phenomenon of emerging timbre, one can foresee the extensive combinatorial complexity embedded in the process of orchestral composition. This complexity have been a major obstacle towards the construction of a scientific basis for the study of orchestration and it remains an empirical discipline taught through the observation of existing examples [5].

Among the different orchestral writing techniques, one of them consists in first laying an harmonic and rhythmic structure in a piano score and then adding the orchestral timbre by spreading the different voices over the various instruments [5]. We refer to this operation of extending a piano draft to an orchestral score as *projective orchestration* [6]. The orchestral repertoire contains a large number of such projective orchestrations (the piano reductions of Beethoven symphonies by Liszt or the *Pictures at an exhibition*, a piano piece by Moussorgsky orchestrated by Ravel and other well-known composers). By observing an example of projective orchestration (Figure 1), we can see that this process involves more than the mere allocation of notes from the piano score across the different instruments. It rather implies harmonic enhancements and timbre manipulations to underline the already existing harmonic and rhythmic structure [3]. However, the visible correlations between a piano score and its orchestrations appear as a fertile framework for laying the foundations of a computational exploration of orchestration.

Statistical inference offers a framework aimed at automatically extracting a structure from observations. These approaches hypothesize that a particular type of data is structured by an underlying probability distribution. The objective is to learn the properties of this distribution, by observing a set of those data. If the structure of the data is efficiently extracted and organized, it becomes then possible to generate novel examples following the learned distribution. A wide range of statistical inference models have been devised, among which *deep learning* appears as a promising field [7, 8]. Deep learning techniques have been successfully applied to several musical applications and neural networks are now the state of the art in most music information retrieval [9–11] and speech recognition [12, 13] tasks. Several generative systems working with symbolic information (musical scores) have also been successfully applied to automatic music composition [14–18] and automatic harmonization [19].

[1] https://qsdfo.github.io/LOP/

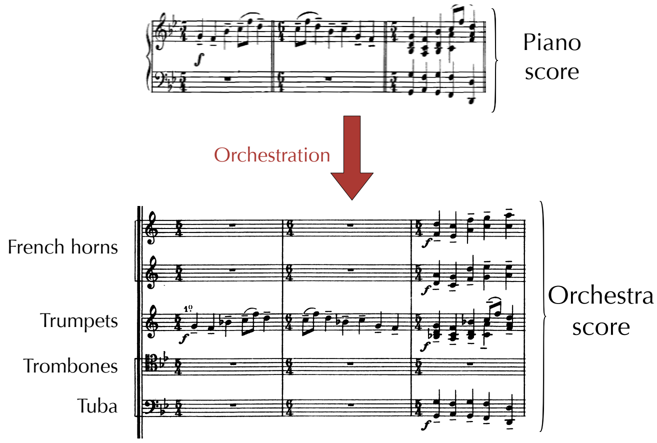


Figure 1. *Projective orchestration.* A piano score is projected on an orchestra. Even though a wide range of orchestrations exist for a given piano score, all of them will share strong relations with the original piano score. One given orchestration implicitly embeds the knowledge of the composer about timbre and orchestration.

Automatic orchestration can actually cover a wide range of different applications. In [20, 21], the objective is to find the optimal combination of orchestral sounds in order to recreate any sonic target. An input of the system is a sound target, and the algorithm explores the different combination using a database of recorded acoustic instruments. The work presented in [22] consists in modifying the style of an existing score. For instance, it can generate a *bossa nova* version of a Beethoven's symphony. To our best knowledge, the automatic projective orchestration task has only been investigated in [23] using a rule-based approach to perform an analysis of the piano score. Note that the analysis is automatically done, but not the allocation of the extracted structures to the different instruments. Our approach is based on the hypothesis that the statistical regularities existing between a corpus of piano scores and their corresponding orchestrations could be uncovered through statistical inference. Hence, in our context, the data is defined as the scores, formed by a series of pitches and intensities for each instrument. The observations is a set of projective orchestrations performed by famous composers, and the probability distribution would model the set of notes played by each instrument conditionally on the corresponding piano score.

It might be surprising at first to rely solely on the symbolic information (scores) whereas orchestration is mostly defined by the spectral properties of instruments, typically not represented in the musical notation but rather conveyed in the signal information (audio recording). However, we make the assumption that the orchestral projection performed by well-known composers effectively took into account the subtleties of timbre effects. Hence, spectrally consistent orchestrations could be generated by uncovering the composers' knowledge about timbre embedded in these scores.

Thus, we introduce the *projective orchestration* task that is aimed at learning models able to generate orchestrations from unseen piano scores. We investigate a class of models called *Conditional RBM* (cRBM) [24]. Conditional models implement a dependency mechanism that seems adapted to model the influence of the piano score over the orchestral score. In order to rank the different models, we establish a novel objective and quantitative evaluation framework, which is is a major difficulty for creative and systems. In the polyphonic music generation field, a predictive task with frame-level accuracy is commonly used by most systems [15, 17, 25]. However, we show that this frame-level accuracy is highly biased and maximized by models that simply repeat their last input. Hence, we introduce a novel event-level evaluation framework and benchmark the proposed models for projective orchestration. Then, we discuss the qualitative aspects of both the models and evaluation framework to explain the results obtained. Finally, we selected the most efficient model and implemented it in a system called *Live Orchestral Piano* (LOP). This system performs real-time projective orchestration, allowing for anyone to play with an orchestra in real-time by simply playing on a MIDI keyboard.

The remainder of this paper is organized as follows. The first section introduces the state of the art in conditional models, in which RBM, cRBM and *Factored-Gated cRBM* (*FGcRBM*) models are detailed. Then, the projective orchestration task is presented along with an evaluation framework based on a event-level accuracy measure. The models are evaluated within this framework and compared to existing models. Then, we introduce *LOP*, the real-time projective orchestration system. Finally, we provide our conclusions and directions of future work.

2. CONDITIONAL NEURAL NETWORKS

In this section, three statistical inference models are detailed. The *RBM*, *cRBM* and *FGcRBM* are presented by increasing level of complexity.

2.1 Restricted Boltzmann Machine

2.1.1 An energy based model

The Restricted Boltzmann Machine (*RBM*) is a graphical probabilistic model defined by stochastic visible units $\boldsymbol{v} = \{v_1, .., v_{n_v}\}$ and hidden units $\boldsymbol{h} = \{h_1, .., h_{n_h}\}$. The visible and hidden units are tied together by a set of weights $\boldsymbol{W}$ following the conditional probabilities

$$p(v_i = 1|\boldsymbol{h}) = \sigma(a_i + \sum_{j=1}^{n_h} W_{ij} h_j) \tag{1}$$

$$p(h_j = 1|\boldsymbol{v}) = \sigma(b_j + \sum_{i=1}^{n_v} W_{ij} v_i) \tag{2}$$

where $\sigma(x) = \frac{1}{1+exp(-x)}$ is the sigmoid function.

The joint probability of the visible and hidden variables is given by

$$p_{model}(\boldsymbol{v}, \boldsymbol{h}) = \frac{\exp^{-E(\boldsymbol{v},\boldsymbol{h})}}{Z} \tag{3}$$

where

$$E(\boldsymbol{v},\boldsymbol{h}) = -\sum_{i=1}^{n_v} a_i v_i - \sum_{i=1}^{n_v}\sum_{j=1}^{n_h} v_i W_{ij} h_j - \sum_{j=1}^{n_h} b_j h_j \quad (4)$$

with $\boldsymbol{\Theta} = \{\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{a}\}$ the parameters of the network.

2.1.2 *Training procedure*

The values of the parameters $\boldsymbol{\Theta}$ of a *RBM* are learned through the minimization of an error function, usually defined as the negative log-likelihood

$$\mathcal{L}(\boldsymbol{\theta}|\mathcal{D}) = \frac{1}{N_{\mathcal{D}}} \sum_{\boldsymbol{v}^{(l)} \in \mathcal{D}} -\ln\left(p(\boldsymbol{v}^{(l)}|\boldsymbol{\theta})\right) \quad (5)$$

where $\mathcal{D}$ is the training dataset and $N_{\mathcal{D}}$ the number of elements that it contains.

The training procedure then modifies the parameters Θ of the model by using the gradient of the error function. However, the gradient of the negative log-likelihood is intractable. Therefore, an approximation of this quantity is obtained by running a Gibbs sampling chain, a procedure known as the Contrastive Divergence (*CD*) algorithm [26,27]. The method alternates between sampling the conditional probabilities of the visible units while keeping the hidden units fixed (Equation 1) and then doing the opposite, until the vector of visible units is close to the real distribution of the model. *CD* provides an adequate approximation for minimizing the loss function and guarantees that the Kullback-Leibler divergence between the model and data distribution is reduced after each iteration [28].

2.2 Conditional models

Conditional models introduce *context units*, which provides an interesting way of influencing the learning. We briefly present the *cRBM* and *FGcRBM* models and redirect interested readers to the original paper [29] for more details.

2.2.1 *Conditional RBM*

In the *cRBM* model, the influence of the context units is implemented by introducing an additive term on the biases a_i and b_j of both visible and hidden units.

$$\tilde{a}_i = a_i + \sum_{k=1}^{n_x} A_{ki} x_k \quad (6)$$

$$\tilde{b}_j = b_j + \sum_{k=1}^{n_x} B_{kj} x_k \quad (7)$$

where $\tilde{\boldsymbol{a}}$ and $\tilde{\boldsymbol{b}}$ are called *dynamic biases* (by opposition to the static biases $\boldsymbol{a}$ and $\boldsymbol{b}$). The additive term is a linear combination of a third set of random variables called context units $\boldsymbol{x} = \{x_1, .., x_k, .., x_{n_x}\}$. By learning the matrices of parameters $\boldsymbol{A}$ and $\boldsymbol{B}$, the context units can stimulate or inhibit the hidden or visible units depending on the input. Hence, this provides a mechanism that can influence the generation based on a particular context. After replacing the static biases by dynamic ones, the conditional distributions (Equation 1), energy function (Equation 4) and training procedure remains the same as for the *RBM*.

2.2.2 *Factored Gated cRBM*

The Factored Gated cRBM (*FGcRBM*) model [29] proposes to extend the cRBM model by adding a layer of feature units $\boldsymbol{z}$, which modulates the weights of the conditional architecture in a multiplicative way. The dynamic biases of the visible and hidden units are defined by

$$\hat{a}_i = a_i + \sum_f \sum_{kl} A_{if} A_{kf} A_{lf} x_k z_l \quad (8)$$

$$\hat{b}_j = b_j + \sum_f \sum_{kl} B_{jf} B_{kf} B_{lf} x_k z_l \quad (9)$$

and the energy function by

$$E(\boldsymbol{v},\boldsymbol{h}) = -\sum_i \tilde{a}_i v_i \quad (10)$$

$$-\sum_f \sum_{ijl} W_{if} W_{jf} W_{lf} v_i h_j z_l - \sum_j \tilde{b}_j h_j \quad (11)$$

This multiplicative influence can be interpreted as a modification of the energy function of the model depending on the latent variables $\boldsymbol{z}$. For a fixed configuration of feature units, a new energy function is defined by the *cRBM* ($\boldsymbol{v}$, $\boldsymbol{h}$, and $\boldsymbol{x}$). Ideally, the three way interactions could be modeled by three dimensional tensors, such as W_{ijl}. To define this, the number of parameters should grow cubically with the number of units. Hence, to reduce the computation load, the three-dimensional tensors are factorized into a product of three matrices by including factor units indexed by f such that $W_{ijl} = W_{if}.W_{jf}.W_{lf}$.

3. PROJECTIVE ORCHESTRATION

In this section, we introduce and formalize the automatic projective orchestration task presented in Figure 1. Then, the piano-roll representation used to process the scores is detailed. Finally, the application of the previously introduced models in this particular context is defined.

3.1 Task formalization

In the general setting, a piano score and an orchestral score can be represented as a sequence of states $P(t)$ and $O(t)$. We consider that the time instants of both sequences are aligned. The projective orchestration task consists in predicting the present orchestral state given the present piano state and the past orchestral states

$$\hat{O}(t) = f\left(P(t), O(t-N), ..., O(t-1)\right) \quad (12)$$

where N is a parameter representing the temporal order (or *horizon*) of the task.

3.2 Data representation

A *piano-roll* representation is used to process both the piano and orchestral scores. This representation is commonly used to model a single polyphonic instrument (Figure 2). Its extension to orchestral scores is obtained straightforwardly by concatenating the *piano-rolls* of each instrument along the pitch dimension in order to obtain a matrix. The rhythmic quantization is defined as the number of time frame in the *piano-roll* per quarter note. Hence, we define the sequence of piano and orchestra states $P(t)$ and $O(t)$ as the sequence of the column vectors formed by the *piano-roll* representations, where t is a discrete time index.

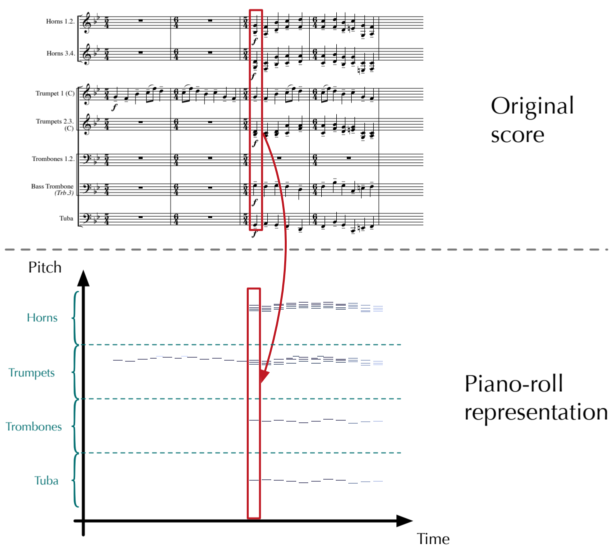


Figure 2. From the score of an orchestral piece, a convenient *piano-roll* representation is extracted. A piano-roll pr is a matrix whose rows represent pitches and columns represent a time frame depending on the time quantization. A pitch p at time t played with an intensity i is represented by $pr(p, t) = i$, 0 being a note off. This definition is extended to an orchestra by simply concatenating the *piano-rolls* of every instruments along the pitch dimension.

3.3 Model definition

The specific implementation of the models for the projective orchestration task is detailed in this subsection.

3.3.1 *RBM*

In the case of projective orchestration, the visible units are defined as the concatenation of the past and present orchestral states and the present piano state

$$\boldsymbol{v} = [P(t), O(t-N), ..., O(t)] \qquad (13)$$

Generating values from a trained *RBM* consists in sampling from the distribution $p(\boldsymbol{v})$ it defines. Since it is an intractable quantity, a common way of generating from a *RBM* is to perform K steps of Gibbs sampling, and consider that the visible units obtained at the end of this process will represent an adequate approximation of the true distribution $p(\boldsymbol{v})$. The quality of this approximation increases with the number K of Gibbs sampling steps.

In the case of projective orchestration, the vectors $P(t)$ and $[O(t-N), ..., O(t-1)]$ are known, and only the vector $O(t)$ has to be generated. Thus, it is possible to infer the approximation $\hat{O}(t)$ by clamping $O(t-N), ..., O(t-1)$ and $P(t)$ to their known values and using the CD algorithm. This technique is known as *inpainting* [26].

However, extensive efforts are made during the training phase to infer values that are finally clamped during the generation phase. Conditional models offer a way to separate the context (past orchestra and present piano) from the data we actually want to generate (present orchestra).

3.3.2 *cRBM and FGcRBM*

In our context, we define the *cRBM* units as

$$\begin{aligned} \boldsymbol{v} &= O(t) \\ \boldsymbol{x} &= [P(t), O(t-N), ..., O(t-1)] \end{aligned}$$

and the *FGcRBM* units as

$$\begin{aligned} \boldsymbol{v} &= O(t) \\ \boldsymbol{x} &= [O(t-N), ..., O(t-1)] \\ \boldsymbol{z} &= P(t) \end{aligned}$$

Generating the orchestral state $O(t)$ can be done by sampling visible units from those two models. This is done by performing K Gibbs sampling steps, while clamping the context units (piano present and orchestral past) in the case of *cRBM* (as displayed in Figure 3) and both the context (orchestral past) and latent units (piano present) in the case of the *FGcRBM* to their known values.

3.3.3 *Dynamics*

As aforementioned, we use binary stochastic units, which solely indicate if a note is on or off. Therefore, the velocity information is discarded. Although we believe that the velocity information is of paramount importance in orchestral works (as it impacts the number and type of instruments played), this first investigation provides a valuable insight to determine the architectures and mechanisms that are the most adapted to this task.

3.3.4 *Initializing the orchestral past*

Gibbs sampling requires to initialize the value of the visible units. During the training process, they can be set to the known visible units to reconstruct, which speeds up the convergence of the Gibbs chain. For the generation step, the first option we considered was to initialize the visible units with the previous orchestral frame $O(t-1)$. However, because repeated notes are very common in the training corpus, the negative sample obtained at the end of the Gibbs chain was often the initial state itself $\hat{O}(t) = O(t-1)$. Thus, we initialize the visible units by sampling a uniform distribution between 0 and 1.

4. EVALUATION FRAMEWORK

In order to assess the performances of the different models presented in the previous section, we introduce a quantitative evaluation framework for the projective orchestration task. Hence, this first requires to define an accuracy measure to compare a predicted state $\hat{O}(t)$ with the ground-truth state $O(t)$ written in the orchestral score. As we experimented with accuracy measures commonly used in the music generation field [15,17,25], we discovered that these are heavily biased towards models that simply repeat their

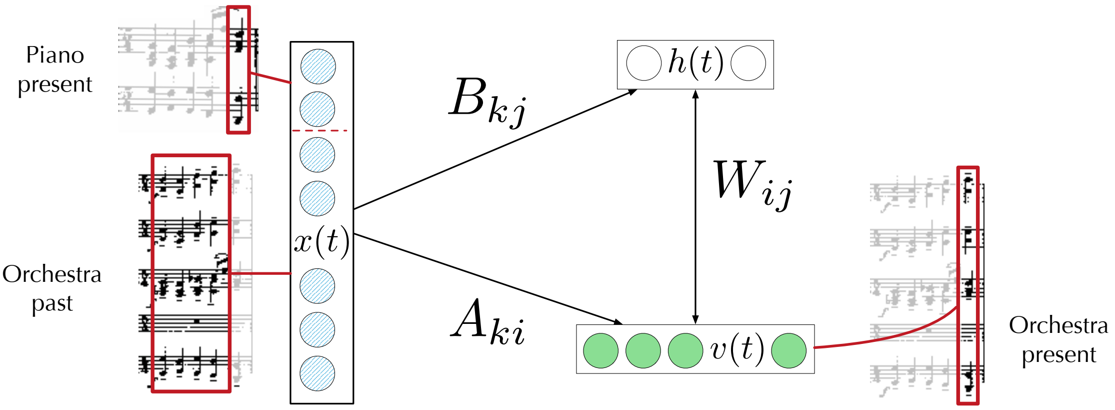


Figure 3. Conditional Restricted Boltzmann Machine in the context of projective orchestration. The visible units $\boldsymbol{v}$ represent the present orchestral frame $O(t)$. The context units $\boldsymbol{x}$ represents the concatenation of the past orchestral and present piano frames $[P(t), O(t-N), ..., O(t-1)]$. After training the model, the generation phase consists in clamping the context units (blue hatches) to their known values and sampling the visible units (filled in green) by running a Gibbs sampling chain.

last input. Hence, we discuss the temporal granularities used to process the scores as piano-rolls and propose two alternatives that we call *frame-level* and *event-level* accuracy measures.

4.1 Accuracy measure

In order to discriminate the performances of different generative models, the ideal evaluation would be to compute the likelihood of an unseen set of data. However, we have seen that this quantity is intractable for probabilistic models such as the *RBM*, *cRBM* and *FGcRBM*. An alternative criterion commonly used in the music generation field [15, 17, 25] is the accuracy measure defined as

$$\text{Accuracy} = \frac{TP(t)}{TP(t) + FP(t) + FN(t)} \tag{14}$$

where the true positives $TP(t)$ is the number of notes correctly predicted, the false positives $FP(t)$ is the number of notes predicted which are not in the original sequence and the false negatives $FN(t)$ is the number on unreported notes.

4.1.1 Temporal granularities

When the accuracy measure defined in Equation 14 is computed for each time index t (termed here *frame-level* granularity) of the *piano-rolls*, this accuracy highly depends on the rhythmic quantization. Indeed, it can be observed on the figure 4 that when the quantization used to compute the *piano-roll* gets finer, an increasing number of successive states in the sequence become identical. As a consequence, a model which simply predicts the state $\hat{O}(t)$ by repeating the previous state $O(t-1)$ gradually becomes the best model as the quantization gets finer.

To alleviate this problem, we rely on an *event-level* granularity, which only assesses the prediction for frames of the piano-roll where an event occurs. We define an event as a time t_e where $\text{Orch}(t_e) \neq \text{Orch}(t_e - 1)$. Hence, the temporal precision of the event-level representation no longer depends on a quantization parameter and the scores are seen as a succession of events with no rhythmic structure. In the general case, this measure should be augmented with the notion of the event durations. However, in our case, the rhythmic structure of the projected orchestral score is imposed by the original piano score.

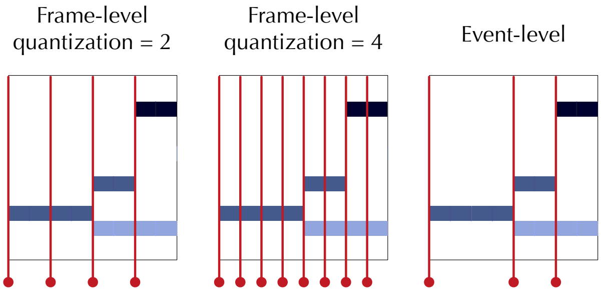


Figure 4. *Frame-level* and *event-level* granularities. In the case of the frame-level granularity (left and middle), we can see that, as the rhythmic quantization gets finer, an increasing number of consecutive frames become identical. However, with the event-level granularity, the bias imposed by the temporal quantization can be alleviated, while still accounting for events that are truly repeated in the score.

In the following section, we discuss the impact of the frame-level and event-level measures on the performance of the different models.

5. RESULTS

5.1 Database

We created a specific MIDI database of piano scores and their corresponding orchestrations performed by famous composers. A total of 223 pairs of piano scores and corresponding orchestrations have been collected and 28 different instruments are represented. The database is freely available [2], along with detailed statistics and the source code to reproduce our results.

Following standard notations, all instruments of the same section are grouped under the same part in the score. For instance, the *violins*, which might be played by several instrumentalists, is written as a single part. Besides, in order to reduce the number of units, we systematically remove, for each instrument, any pitch which is never played in the database. Hence, the dimension of the orchestral vector is reduced from 3584 to 1220.

The dataset is split between train, validation and test sets of files that represent respectively 80%, 10% and 10% of the full set. For reproducibility, the different sets we used are provided as text files on the companion website.

5.2 Quantitative evaluation

We evaluate five different models on the projective orchestration task. The first model is a random generation of the orchestral frames from a Bernoulli distribution of parameter 0.5 in order to evaluate the complexity of the task. The second model predicts an orchestral frame at time t by repeating the frame at time $t-1$, in order to evaluate the *frame-level* bias. These two naive models constitute a baseline against which we compare the *RBM*, *cRBM* and *FGcRBM* models.

The complete implementation details and hyper-parameters can be found on the companion website.

Model	Frame-level accuracy (Q = 4)	Frame-level accuracy (Q = 8)	Event-level accuracy
Random	0.73	0.73	0.72
Repeat	61.79	76.41	50.70
RBM	7.67	4.56	1.39
cRBM	5.12	34.25	27.67
FGcRBM	33.86	43.52	25.80

Table 1. Results of the different models for the projective orchestration task based on frame-level accuracies with a quantization of 4 and 8 and event-level accuracies.

The results are summarized in Table 1. In the case of frame-level accuracies, the *FGcRBM* provides the highest accuracy amongst probabilistic models. However, the repeat model remains superior to all models in that case. If we increase the quantization, we see that the performances of the repeat model increase considerably. This is also the case for the *cRBM* and *FGcRBM* models as the predictive objective becomes simpler.

The RBM model obtains poor performances for all the temporal granularities and seems unable to grasp the underlying structure. Hence, it appears that the conditioning is of paramount importance to this task. This could be explained by the fact that a dynamic temporal context is one of the foremost properties in musical orchestration.

The *FGcRBM* has slightly worse performances than the *cRBM* in the event-level framework. The introduction of three ways interactions might not be useful or too intricate in the case of projective orchestration. Furthermore, disentangling the interactions of the piano and the orchestra against the influence of the temporal context might not be clearly performed by the factored model because of the limited size of the database. Nevertheless, it seems that conditional probabilistic models are valid candidates to tackle the projective orchestration task.

Even with the event-level accuracy, the repeat model still obtains a very strong score, which further confirms the need to devise more subtle accuracy measures. This result directly stems from the properties of orchestral scores, where it is common that several instruments play a sustained background chord, while a single solo instrument performs a melody. Hence, even in the event-level framework, most of the notes will be repeated between two successive events. This can be observed on the database section of the companion website.

5.3 Qualitative analysis

In order to allow for a qualitative analysis of the results, we provide several generated orchestrations from different models on our companion website. We also report the different accuracy scores obtained by varying the hyper-parameters. In this analysis, it appeared that the number of hidden units and number of Gibbs sampling steps are the two crucial quantities. For both the *cRBM* and *FGcRBM* models, best results are obtained when the number of hidden units is over 3000 and the number of Gibbs sampling steps larger than 20.

We observed that the generated orchestrations from the *FGcRBM* might sound better than the ones generated by the *cRBM*. Indeed, it seems that the *cRBM* model is not able to constraint the possible set of notes to the piano input. To confirm this hypothesis, we computed the accuracy scores of the models while setting the whole piano input to zero. By doing this, we can evaluate how much a model rely on this information. The score for those *corrupted* inputs considerably dropped for the *FGcRBM* model, from 25.80% to 1.15%. This indicates that the prediction of this model heavily rely on the piano input. Instead, the corrupted score of the *cRBM* drops from 27.67% to 17.80%. This shows that the piano input is not considered as a crucial information for this model, which rather performs a purely predictive task based on the orchestral information.

5.4 Discussion

It is important to state that we do not consider this prediction task as a reliable measure of the creative performance of different models. Indeed, predicting and creating are two fairly different things. Hence, the predictive evaluation framework we have built does not assess the generative ability of a model, but is rather used as a selection criterion among different models. Furthermore, this provides an interesting *auxiliary task* towards orchestral generation.

6. LIVE ORCHESTRAL PIANO (LOP)

We introduce here the *Live Orchestral Piano* (LOP), which allows to compose music with a full classical orchestra in

[2] `https://qsdfo.github.io/LOP/database`

real-time by simply playing on a MIDI piano. This framework relies on the knowledge learned by the models introduced in the previous sections in order to perform the projection from a piano melody to the orchestra.

6.1 Workflow

The software is implemented on a client/server paradigm. This choice allows to separate the orchestral projection task from the interface and sound rendering engine. That way, multiple interfaces can easily be implemented. Also, it should be noted that separating the computing and rendering aspects on different computers could allow to use high-quality and CPU-intensive orchestral rendering plugins, while ensuring the real-time constraint on the overall system (preventing degradation on the computing part). The complete workflow is presented in Figure 5.

The user plays a melody (single notes or chords sequences) with a MIDI keyboard, which is retrieved inside the interface. The interface has been developed in Max/Msp, to facilitate both the score and audio rendering aspects. The interface transmits this symbolic information (as a variable-length vector of active notes) via OSC to the MATLAB server. This interface also performs a transcription of the piano score to the screen, by relying on the *Bach* library environment. The server uses this vector of events to produce an 88 vector of binary note activations. This vector is then processed by the orchestration models presented in the previous sections in order to obtain a projection of a specific symbolic piano melody to the full orchestra. The resulting orchestration is then sent back to the client interface which performs both the real-time audio rendering and score transcription. The interface also provides a way to easily switch between different models, while controlling other hyper-parameters of the sampling

7. CONCLUSION AND FUTURE WORKS

We have introduced a system for real-time orchestration of a midi piano input. First, we formalized the projective orchestration task and proposed an evaluation framework that could fit the constraints of our problem. We showed that the commonly used *frame-level* accuracy measure is highly biased towards repeating models and proposed an *event-level* measure instead. Finally, we assessed the performance of different probabilistic models and discussed the better performances of the FGcRBM model.

The conditional models have proven to be effective in the orchestral inference evaluation framework we defined. The observations of the orchestration generated by the models tend to confirm these performance scores, and support the overall soundness of the framework. However, we believe that the performances can be further improved by mixing conditional and recurrent models.

The most crucial improvement to our system would be to include the dynamics of the notes. Indeed, many orchestral effects are directly correlated to the intensity variations in the original piano scores. Hence, by using binaries units, an essential information is discarded. Furthermore, the sparse representation of the data suggests that a more compact distributed representation might be found. Lowering the dimensionality of the data would greatly improve the efficiency of the learning procedure. For instance, methods close to the word-embedding techniques used in natural language processing might be useful [30].

The general objective of building a generative model for time series is one of the currently most prominent topic for the machine learning field. Orchestral inference sets a slightly more complex framework where a generated multivariate time series is conditioned by an observed time series (the piano score). Finally, being able to grasp long-term dependencies structuring orchestral pieces appears as a promising task.

8. ACKNOWLEDGEMENTS

This work has been supported by the *NVIDIA GPU Grant* program. The authors would like to thank Denys Bouliane for kindly providing a part of the orchestral database.

9. REFERENCES

[1] C. Koechlin, *Traité de l'orchestration*. Éditions Max Eschig, 1941.

[2] N. Rimsky-Korsakov, *Principles of Orchestration*. Russischer Musikverlag, 1873.

[3] S. McAdams, "Timbre as a structuring force in music," in *Proceedings of Meetings on Acoustics*, vol. 19, no. 1. Acoustical Society of America, 2013, p. 035050.

[4] D. Tardieu and S. McAdams, "Perception of dyads of impulsive and sustained instrument sounds," *Music Perception*, vol. 30, no. 2, pp. 117–128, 2012.

[5] W. Piston, *Orchestration*. New York: Norton, 1955.

[6] P. Esling, "Multiobjective time series matching and classification," Ph.D. dissertation, IRCAM, 2012.

[7] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 8, pp. 1798–1828, 2013.

[8] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 05 2015. [Online]. Available: http://dx.doi.org/10.1038/nature14539

[9] E. J. Humphrey, J. P. Bello, and Y. LeCun, "Moving beyond feature design: Deep architectures and automatic feature learning in music informatics." in *ISMIR*. Citeseer, 2012, pp. 403–408.

[10] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Unsupervised learning of hierarchical representations with convolutional deep belief networks," *Communications of the ACM*, vol. 54, no. 10, pp. 95–103, 2011.

[11] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent, "Audio chord recognition with recurrent neural networks." in *ISMIR*. Citeseer, 2013, pp. 335–340.

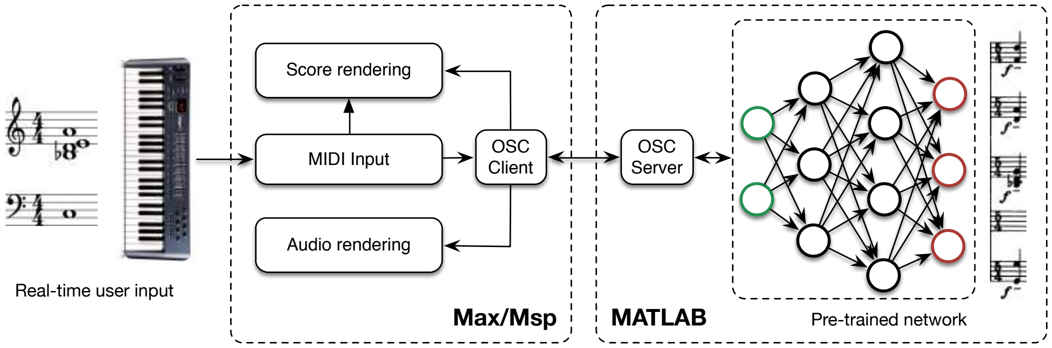


Figure 5. Live orchestral piano (LOP) implementation workflow. The user plays on a MIDI piano which is transcribed and sent via OSC from the Max/Msp client. The MATLAB server uses this vector of notes and process it following the aforementioned models in order to obtain the corresponding orchestration. This information is then sent back to Max/Msp which performs the real-time audio rendering.

[12] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[13] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *CoRR*, vol. abs/1609.03499, 2016. [Online]. Available: http://arxiv.org/abs/1609.03499

[14] D. Eck and J. Schmidhuber, "Finding temporal structure in music: Blues improvisation with lstm recurrent networks," in *Neural Networks for Signal Processing, 2002. Proceedings of the 2002 12th IEEE Workshop on.* IEEE, 2002, pp. 747–756.

[15] V. Lavrenko and J. Pickens, "Polyphonic music modeling with random fields," in *Proceedings of the eleventh ACM international conference on Multimedia.* ACM, 2003, pp. 120–129.

[16] S. Bosley, P. Swire, R. M. Keller *et al.*, "Learning to create jazz melodies using deep belief nets," 2010.

[17] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent, "Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription," *arXiv preprint arXiv:1206.6392*, 2012.

[18] D. Johnson, "hexahedria," http://www.hexahedria.com/2015/08/03/composing-music-with-recurrent-neural-networks/, 08 2015.

[19] F. Sun, "Deephear - composing and harmonizing music with neural networks," http://web.mit.edu/felixsun/www/neural-music.html.

[20] G. Carpentier, D. Tardieu, J. Harvey, G. Assayag, and E. Saint-James, "Predicting timbre features of instrument sound combinations: Application to automatic orchestration," *Journal of New Music Research*, vol. 39, no. 1, pp. 47–61, 2010.

[21] P. Esling, G. Carpentier, and C. Agon, "Dynamic musical orchestration using genetic algorithms and a spectro-temporal description of musical instruments," *Applications of Evolutionary Computation*, pp. 371–380, 2010.

[22] F. Pachet, "A joyful ode to automatic orchestration," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 8, no. 2, p. 18, 2016.

[23] E. Handelman, A. Sigler, and D. Donna, "Automatic orchestration for automatic composition," in *1st International Workshop on Musical Metacreation (MUME 2012)*, 2012, pp. 43–48.

[24] G. W. Taylor, G. E. Hinton, and S. T. Roweis, "Modeling human motion using binary latent variables," in *Advances in neural information processing systems*, 2006, pp. 1345–1352.

[25] K. Yao, T. Cohn, K. Vylomova, K. Duh, and C. Dyer, "Depth-gated LSTM," *CoRR*, vol. abs/1508.03790, 2015. [Online]. Available: http://arxiv.org/abs/1508.03790

[26] A. Fischer and C. Igel, *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications: 17th Iberoamerican Congress, CIARP 2012, Buenos Aires, Argentina, September 3-6, 2012. Proceedings.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, ch. An Introduction to Restricted Boltzmann Machines, pp. 14–36. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-33275-3_2

[27] G. Hinton, "A practical guide to training restricted boltzmann machines," *Momentum*, vol. 9, no. 1, p. 926, 2010.

[28] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[29] G. W. Taylor and G. E. Hinton, “Factored conditional restricted boltzmann machines for modeling motion style,” in *Proceedings of the 26th annual international conference on machine learning*. ACM, 2009, pp. 1025–1032.

[30] R. Kiros, Y. Zhu, R. R. Salakhutdinov, R. Zemel, R. Urtasun, A. Torralba, and S. Fidler, “Skip-thought vectors,” in *Advances in Neural Information Processing Systems*, 2015, pp. 3276–3284.

A SINGING INSTRUMENT FOR REAL-TIME VOCAL-PART ARRANGEMENT OF MUSIC AUDIO SIGNALS

Yuta Ojima[1], **Tomoyasu Nakano**[2], **Satoru Fukayama**[2], **Jun Kato**[2], **Masataka Goto**[2],
Katsutoshi Itoyama[1] **and Kazuyoshi Yoshii**[1]
[1] Graduate School of Informatics, Kyoto University, Japan
[2] National Institute of Advanced Industrial Science and Technology (AIST)
{ojima, itoyama, yoshii}@sap.ist.i.kyoto-u.ac.jp
{t.nakano, s.fukayama, jun.kato, m.goto}@aist.go.jp

ABSTRACT

This paper presents a live-performance system that enables a user to arrange the vocal part of music audio signals and to add a harmony or troll part by modifying the original vocal part. The pitches and rhythms of the original singing voices can be manipulated in real time while preserving the lyrics. This leads to an interesting experience in which the user can feel as if the original singer is directed to sing a song as the user likes. More specifically, a user is asked to play a MIDI keyboard according to the user's favorite melody. Since the vocal part is divided into short segments corresponding to musical notes, those segments are played back one by one from the beginning such that the pitch and duration of each segment matches a key push by the user. The functions needed for this system are singing voice separation, vocal-part segmentation, key-to-segment association, and real-time pitch modification. We propose three kinds of key-to-segment association to restrict the degree of freedom in manipulation of pitches and durations and to easily generate a harmony or troll part. Subjective experiments showed the potential of the proposed system.

1. INTRODUCTION

We enjoy music not only in a passive way but also in an active way, such as arranging or playing existing songs with a musical instrument. By arranging and playing songs, we can substitute for any part in the music a favorite melody or rhythm pattern. Today, more and more people arrange a part in an existing song, play the arranged part overlaying the original song, and upload the performance to video hosting services such as YouTube.

Arrangement systems for existing songs have already been proposed [1, 2]. These systems help a user arrange an instrumental part in an existing song. Similarly, there exists a situation we hope to arrange a vocal part in existing songs. In fact, many amateur singers uploads vocal covers called "Me singing" on video hosting service [1]. If one does not sing well enough to substitute his or her singing voice for the original singing voice, one needs to make use of a singing voice synthesis system such as *VOCALOID* [3]. Such a system, however, requires lyrics and pitch information in advance, and this information is not easy to get.

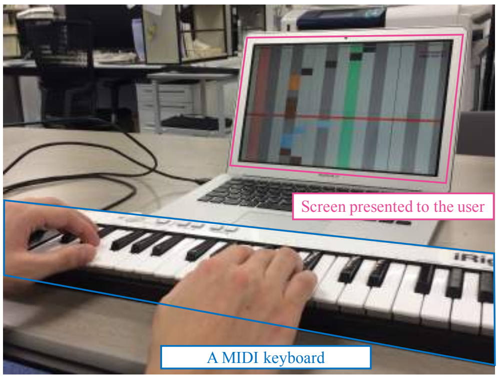


Figure 1. Example of how the proposed system is used. The original vocal part is visualized on a screen, and a user modifies it by using a MIDI keyboard.

This paper proposes a vocal arrangement system that directly manipulates the original singing voice extracted from existing songs in real time (Fig. 1) [2]. There are two advantages in using the original singing voice. The first is that the arrangement is realized without preparing lyrics. The second is that the individuality of the singer is preserved after the arrangements. By preserving the individuality, this system provides an experience as if the original artist sang in a way as a user likes. The arrangements we focus on are pitch modification and changes of onset/offset timing, and we need our system to let users manipulate the pitches, onset times, and durations of notes simultaneously and intuitively. We therefore use a MIDI keyboard as the user interface. This lets users perform singing voice arrangement as if they were playing a piano.

To realize this system, we tackle three problems: estimation and visualization of musical notes in a vocal part, appropriate assignment of a musical note to the key played by a user, and real-time resynthesis of the singing voice after its pitch modified. For the first problem, we first extract an F0 trajectory of the singing voice by using robust principle component analysis (RPCA) [4]. Then, we estimate mu-

[1] More than 98 million "Me singing" videos are uploaded on a popular video sharing service *YouTube*.

[2] A demo video of the proposed system is available online: http://sap.ist.i.kyoto-u.ac.jp/members/ojima/smc2017/index.html

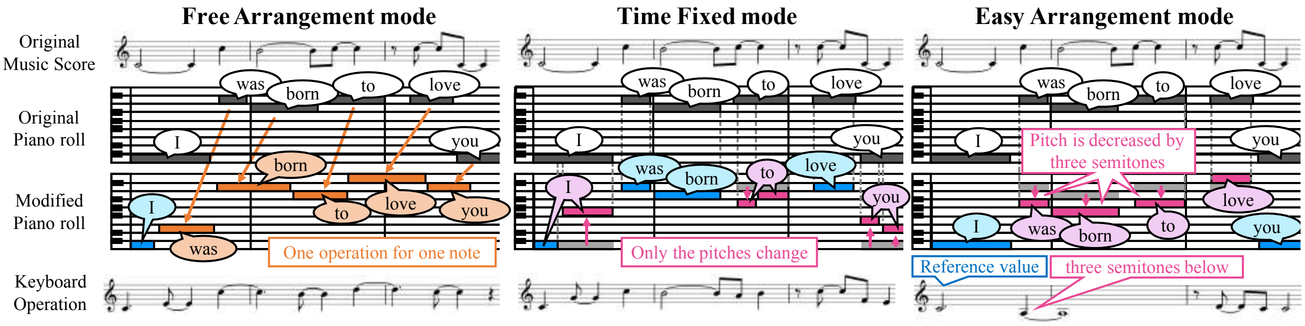


Figure 2. The description of three assignment modes. Each balloon represents the lyrics assigned to a note. In the Free Arrangement mode, the note in the original vocal part is assigned one by one from the beginning for every key operation. In the Time Fixed mode, only the pitches can be changed and the onset/offset timings are not changed. In the Easy Arrangement mode, a user modifies the pitch by specifying its interval relative to the original melody.

sical notes from the F0 trajectory to visualize pitch information by using a Hidden Markov Model (HMM) [5]. For the second problem, we prepare three assignment modes to make it easy to reflect users' various intentions for arrangement (Fig. 2). Since these assignment modes can be switched during the performance, a user can perform as they likes. For the third problem, we use the pitch synchronous overlap and add (PSOLA) algorithm because the algorithm is simple enough to work in real time. Since the arrangement is performed in real time using a MIDI keyboard, this system can be used for a live performance as a DJ plays a song.

2. RELATED WORK

In this section, we consider work related to different aspects of our vocal arrangement system. We first review various music applications and then we focus on arrangement systems because our system aims to arrange an existing music by using signal processing methods. At the end of this section, we review the signal processing methods that are deeply relevant to our system.

2.1 Music Applications

Our system supports active music listening by visualizing the vocal part and by replacing the melody with the one a user likes. Regarding active music listening, several studies have been made. Goto *et al.* [6] proposed a Web service called *Songle* that enriches music listening experience by visualizing the music structure, beat structure, melody line, and chords. Mauch *et al.* [7] proposed an interface called *Song Prompter* that displays the structure of a song, chords, and lyrics without preparing musical scores or manually aligning the lyrics and chords with the audio. Nakra *et al.* [8] designed an interactive conducting system that enables a user to change the tempo and dynamics of an orchestral recording by *Wii Remote*.

On the other hand, another aspect of our system is that it is a music generation system. Many music generation support systems have been proposed. Simon *et al.* [9] proposed one that generates a chord progression to accompany the given vocal melody. Using this system, users who are not familiar with musicography can easily try composing a musical piece. McVicar *et al.* [10] proposed a system that composes a guitar tablature when a chord progression is given as an input. Since this system needs tablature and chords given in advance for training, the generated guitar tablature may vary according to the data used in training. This contributes to the generation of guitar tablature that reflects individuality underlying in the training data. Methods that generate the harmony part have also been proposed. Yi *et al.* [11] proposed a method to generating four-part harmony automatically when a melody is given. Dannenberg *et al.* [12] focused on the music performance generated by cooperating with computers. They aim to enable musicians to incorporate computers into their performances by creating computer performers that play music along with music.

Our system is also regarded as an automatic accompaniment system for a vocal part generated by a user. Several such accompaniment systems have been proposed. Originally, Dannenberg [13] proposed an algorithm for finding the best match between an input music score and a solo performance with allowing some mistakes. Using that algorithm, he designed a system that produced an accompaniment in real time. Cont [14] also proposed an automatic accompaniment system. He used hidden hybrid Markov/semi-Markov models to estimate the current score position and dynamics of a tempo from input audio and a symbolic music score.

2.2 Arrangement Systems for Existing Songs

Many systems that can arrange existing songs have been proposed [1, 2, 15]. Yasuraoka *et al.* [2] proposed a music manipulation system that can change the timbre and phrase of a certain musical instrument in polyphonic music when the corresponding musical score is given. They define a musical-tone model to extract the target musical instrument. In this system, a user is asked to show his or her desirable arrangement in advance in the form of a music score. Tsuzuki *et al.* [16] proposed a system that extracts multiple vocal parts from existing songs and uses them to make a mashup. That system extracts multiple vocal parts of different singers singing the same song. The extracted voices are then overlapped on one shared accompaniment.

Some systems that assist the manipulation of an original instrumental part in real time have also been proposed. Yoshii *et al.* [1] proposed an arrangement system for a drum part. It enables a user to control the volume and timbre of a drum part and to edit drum patterns in real time by using a drum sound recognition method and beat-tracking. Yamamoto *et al.* [15] propose a user interface for synthesizing a singing voice for a live improvisational performance. This interface enables real-time singing synthesis when lyrics information is given in advance.

2.3 Signal Processing Methods

We review relevant studies on vocal-and-accompaniment source separation (VASS), musical note estimation methods from F0 trajectories, and pitch modification in this section.

2.3.1 Source Separation

Methods for vocal-and-accompaniment source separation (VASS) have intensively been studied [4, 17, 18]. Rafii *et al.* [17] proposed a repeated pattern extraction technique (REPET). They focus on repetitive structures in music to extract a vocal part since accompaniments are repeated while a vocal part is not repeated. Huang *et al.* [18] realized singing voice separation by using robust principle component analysis (RPCA). They model an accompaniment part with repetitive structures as a low-rank component and model a vocal part as a sparse component. Ikemiya *et al.* [4] combined RPCA and pitch estimation of singing voices to realize VASS. They focus on the mutual dependency of singing voice separation and F0 estimation, and they improve singing voice separation in a unified framework.

2.3.2 Musical Note Estimation from F0 Trajectories

There have been several attempts to estimate musical notes when an F0 trajectory is given. In note estimation, discrete values with the interval of semitones are estimated from a continuous F0 trajectory. In *Songle* [6] these discrete values are estimated by using a majority-vote method for every time unit (*e.g.*, half beat). Laaksonen *et al.* [19] proposed a method of creating a symbolic melody when a chord progression and audio data are given. Ryynänen *et al.* [20] considered some states within one musical note (*e.g.*, vibrato and overshooting) and represented the inter-state transitions by using a hidden Markov model (HMM). Nishikimi *et al.* [5] represented the generative process of an F0 trajectory from underlying musical notes by using a Bayesian HMM.

2.3.3 Pitch Modification Algorithms

Pitch modification is a major requirement in music arrangement and has been examined in many studies [21]. In the time domain, pitch modification that preserves the original duration is achieved by time stretching and compression/expansion of the waveform. Many algorithms for time stretching have been proposed, and the simplest is that of Roucos *et al.* [22]. In their method, the waveform is divided into overlapping segments of a fixed length, and then they are shifted and added to realize time stretching. Hamon *et al.* [23] expanded this algorithm to make use of pitch information. In their method, the length of divided segments is determined according to the original pitch. Some of these segments are repeated or discarded to realize time stretching or time compression.

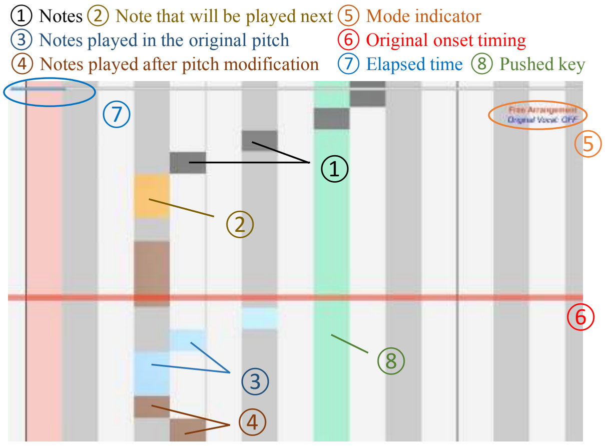

① Notes ② Note that will be played next ⑤ Mode indicator
③ Notes played in the original pitch ⑥ Original onset timing
④ Notes played after pitch modification ⑦ Elapsed time ⑧ Pushed key
⑦
Free Arrangement
Original Vocal: OFF
⑤
①
②
⑥
⑧
③
④

Figure 3. A screenshot of the proposed system. Each note comes down from above and is colored corresponding to its status. The current playing mode is indicated in the upper right corner. The pushed key is highlighted green.

3. USER INTERFACE

This section gives detailed descriptions of the proposed system and its user interface (Fig. 3). Section 3.1 explains the user experience which this system provides. Section 3.2 describes the user interface in detail.

3.1 Overview of the Proposed System

We propose a vocal arrangement system that enables users to manipulate a vocal part as if they were playing an instrument by using a MIDI keyboard. We focus on the following functions:

- Replacing the melody of the original vocal part with a melody as the user likes
- Adding a harmonic part
- Adding a troll part (*i.e.* an auxiliary part that follows a main part)

On the occasion of these arrangements, users change the features of the vocal part (*i.e.* the pitches, onset times, and durations) simultaneously. To treat these manipulations intuitively, we use a MIDI keyboard as the user interface. The target users are therefore the persons who can play the piano.

In this system, the vocal part is divided at the point where the pitch changes and is arranged. We henceforth refer to each divided segment as a *note*. When a key of a MIDI keyboard is pushed, the system first determines a note that is a target of his or her key operation. We henceforth refer to this note as an *target note*. Then the pitch and the duration of the target note are modified corresponding to user's key operation. We prepare three arrangement modes to determine the target note and to specify its pitch: *Free Arrangement mode*, *Time Fixed mode* and *Easy Arrangement mode* (Fig. 2).

Free Arrangement mode The note which has the earliest onset time in the notes that have not played by a user yet is estimated as the target note. The pitch of the target note is shifted to the pitch of a key played by a user. While key operation is needed for every musical note, a user can change the onset time and the pitches no matter how far they are from the original ones.

Time Fixed mode The note being played in the original vocal part at the moment a user pushes the key is estimated as the target note. While the onset times and the durations are unchangeable, a user can change the pitch at the middle point of a note in this mode. The pitch of the target note is shifted to the pitch of a key played by a user as in the Free Arrangement mode. Moreover, a note is automatically assigned to music one by one even though a user keeps a key pushed for multiple notes. A user therefore does not have to push a key again if they wants to play multiple continuous notes in the same pitch.

Easy Arrangement mode The target note is determined in the same way as in the Time Fixed mode. In this mode, how the system modifies the pitch of the target note is specified by the number of semitones between a key pushed by a user and C4 (*e.g.*, if a user pushes E4, the pitch of the target note is increased by four semitones).

These three arrangement modes are prepared to cope with several different motivations for arrangement. When a user wants to manipulate all the vocal part as they likes, the Free Arrangement mode is suitable. On the other hand, when a user wants to manipulate a portion of the vocal part and leaves the rest as it is, the Time Fixed mode or the Easy Arrangement mode is suitable since with it the user does not have to make keyboard manipulations for notes that they wants to leave unchanged. When a user wants to add a harmonic part, the Easy Arrangement mode may be helpful because typical harmonic parts are played in a certain pitch relative to the original melody line (*e.g.*, third below) and the onset times and durations in harmonic parts are the same as ones in the original melody. Since a user can switch these modes at a keyboard during the performance, thereby more easily accessing the functions they need to accomplish the arrangement, the system can handle the different kinds of arrangement in a unified framework.

Furthermore, since some users want to eliminate the original vocal part while others just want to add a harmonic or troll part, leaving the original vocal part unchanged, the interface lets a user toggle the original singing voice on and off. This operation is also controllable during the performance. A user can also pause or resume the song by using a MIDI keyboard. Since a user needs to keep his or her eyes on the screen and cannot spare the attention needed to operate any equipments other than a MIDI keyboard, all the operations during the performance are controlled using only a MIDI keyboard.

3.2 Design of the Screen

The requirements for the screen presented to the user are as follows:

- The pitches and durations of musical notes in the original vocal part are shown in the way that they are intuitively and immediately evident to the user.

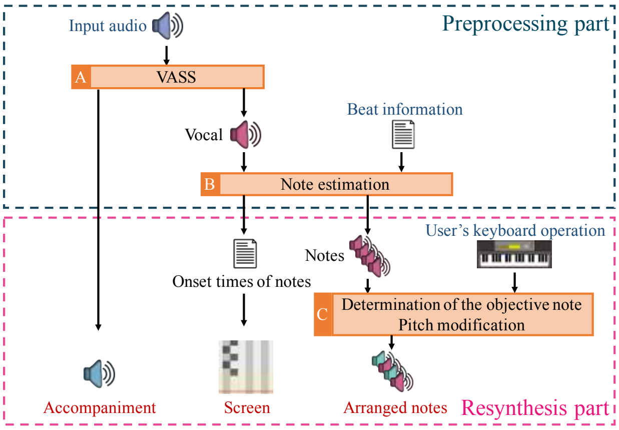


Figure 4. Overview of the proposed system: Inputs are indicated by blue letters and outputs are indicated by red letters.

- The response to a user's operation is displayed immediately.
- Users can predict the effect of their operation before performing the operation.

The first requirement is imposed because users sometimes refer to the original melody during their arrangement. The second requirement is imposed to make arrangement smooth and stress-free for users. The last requirement is imposed to let users get the modified performances they expect.

We designed the screen to meet these requirements (Fig.3). To meet the first requirement, we design the screen to have the horizontal direction indicating the pitches of the original vocal part. Moreover, we colored each line white or gray in order to represent white and black keys on a MIDI keyboard. This layout is like the one of a MIDI keyboard. Moreover, the line that indicates C4 (MIDI note number 60) is shown in red to make the octave of keys explicit. Musical notes in the original vocal part are visualized as rectangles coming down from above. This design is similar to the one of existing musical games, so users easily understand the presented screen.

To meet the second requirement, the color of the line corresponding to the key played by a user is changed to light green and the colors of notes that have already been played by user are changed into light blue or brown. By these changes in colors, the interface lets users understand the effect of their key operation and the pushed key without looking at their hands.

To meet the last requirement, the target note at the moment is shown in orange. This helps prevent unexpected manipulations.

Since the number of pitches that appear in the original vocal part varies according to the song, the number of lines shown in the screen changes according to the highest pitch and the lowest pitch in the song. The elapsed time is visualized at the top of the window.

4. SYSTEM IMPLEMENTATION

This section describes the signal processing techniques used in the system. As shown in Fig.4, the system is roughly divided into two parts: a *preprocessing part* and a *resynthesis part*. In the preprocessing part, a singing voice is extracted from the song. This is enabled by vocal-and-accompaniment source separation (VASS). Then the timings that the pitch switches are estimated and the singing voice is divided at those timings. This is enabled by note estimation from an F0 trajectory. In the resynthesis part, a vocal part is manipulated and resynthesized according to the user's operations. The pitch of the note is modified to the one specified by the user, and then a modified singing voice is re-synthesized.

4.1 Vocal-and-Accompaniment Source Separation

We used the VASS method proposed by Ikemiya *et al.* [4] to extract a singing voice ("A" in Fig. 4) because it not only extracts a singing voice but also estimates an F0 trajectory that can be used to estimate musical notes. This method first represents the given music spectrogram as the sum of a low-rank matrix and a sparse matrix by applying RPCA. Since the singing voice is predominant in the sparse matrix, we extract the sparse matrix by applying a binary-mask and treat the extracted sparse matrix as a *rough* vocal spectrogram. Then the most likely F0 trajectory of a rough vocal spectrogram is estimated by subharmonic summation. Using this F0 trajectory, a harmonic mask that passes only the harmonic partials of estimated F0s is made. Finally it is integrated with the binary-mask mentioned above to extract a vocal spectrogram and an accompaniment spectrogram from the music spectrogram. The singing voice and the accompaniment are then synthesized using these spectrograms.

4.2 Note Estimation from an F0 Trajectory

We used an HMM-based method [5] for musical note estimation ("B" in Fig. 4). In this method, the process generating an F0 trajectory from underlying musical notes is modeled by using a Bayesian HMM. When beat information is given in advance, the pitches with the interval of semitones are estimated at a sixteenth-note level. Using this information, the notes of the vocal part are displayed to a user and the extracted singing voice is divided by note durations.

4.3 Pitch Modification

After the assignment of the note to the key, the pitch of the note is modified to the pitch of the key ("C" in Fig. 4). As noted in Section 2.3.3, several pitch modification algorithms have been proposed. We used PSOLA algorithm (Fig. 5) because it is simple enough to work in real time. Since it uses pitch information, the sound quality is also improved compared to the simplest SOLA algorithm. In this algorithm, first the waveform is divided by twice the length of its estimated wavelength, which is calculated from the estimated pitch. Then the divided segments are rearranged for time stretching, in which some segments are repeated or discarded if needed. Each segment is overlapped with the neighboring segments. After the rearrangement, overlapped segments are weighted by a window function and added. Finally, the rearranged segments are compressed or expanded to the original duration.

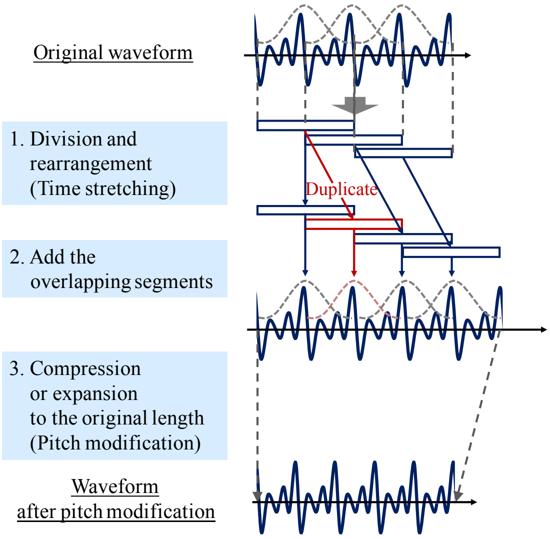

Original waveform
1. Division and rearrangement (Time stretching)
Duplicate
2. Add the overlapping segments
3. Compression or expansion to the original length (Pitch modification)
Waveform after pitch modification

Figure 5. PSOLA algorithm: First the waveform is divided by twice the length of its wavelength and rearranged to realize time stretching, and then it is compressed or expanded to realize pitch modification.

5. EXPERIMENT

This section reports the subjective experiments conducted to evaluate the effectiveness of this system.

5.1 Experimental Conditions

We asked four subjects to use this system on RWC-MDB-P-2001 No.7 in the RWC music database (popular) [24]. In the experiment, we first showed the subjects the performance made by one of the authors as a tutorial. Then, we observed how each subject used the system and asked for his or her opinions in terms of the functionality of the system, the user interface, and the signal processing. We also asked the subject to evaluate the system in terms of the evaluation items shown below in a 5-point Likert scale (*i.e.*, strongly disagree, disagree, neutral, agree, strongly agree).

1. The Free Arrangement mode was easy to use.
2. The Time Fixed mode was easy to use.
3. The Easy Arrangement mode was easy to use.
4. The displayed information was understandable.
5. The lyrics were distinguishable after the pitch modification.
6. The pitches were modified as intended.
7. The individuality of the singer was preserved after the pitch modification.

In this experiment, we used as the notes shown on the screen the correct notes in the experiment. Finally, we used as the notes shown on the screen the notes estimated from an F0 trajectory to examine whether or not the precision of notes shown in the screen affects the ease of performance.

The subjects were four university students (three males and one female). All the subjects had played the piano for more than five years. Before the experiment, we asked the subject to listen to the original song.

	Strongly disagree	Disagree	Nautral	Agree	Strongly agree
Free Arrangement mode was easy to use	1	2	1		
Time Fixed mode was easy to use		1		3	
Easy Arrangement mode was easy to use	1			1	2
Displayed information was understandable			2	2	
Lyrics were understandable		1		2	1
Pitches were modified as intended			1	2	1
Individuality was preserved	2		1	1	

Figure 6. Evaluation of the proposed system. Each number represents the number of subjects who selected the corresponding item.

5.2 Experimental Results

In the experiment, the subjects tried multiple modes implemented on the system and the Time Fixed mode was mostly used. Most subjects used the Easy Arrangement mode to add a harmonic part to the original vocal part. In the Free Arrangement mode, sometimes it seemed hard for the subjects to comprehend the note they were playing. The result of the system evaluation is shown in Fig. 6. According to the result, the Time Fixed mode was popular among all the users. Moreover, the Easy Arrangement mode was mostly highly evaluated. On the other hand, the Free Arrangement mode was lowly evaluated due to its difficulties. As for the signal processing, the pitches were modified as the users intended, and the lyrics were recognizable after the pitch modification. On the other hand, this result indicates that the individuality of the singer was lost after the pitch modification. The remarkable opinions we obtained after the experiment are shown below.

The following opinions were obtained on the function:

- The system enabled me to get the sense of manipulating the vocal as if I were playing an instrument.
- It was fun to manipulate the vocal part as if the singer hit a wrong note.
- The assignment mode could be switched seamlessly.
- This system may be useful for practicing piano.
- *The easy mode* was useful for adding the expressions such as glissando or vibrato.

The obtained opinions show the effectiveness of the system as a kind of musical instruments for a singing voice.

The following opinions were obtained on the user interface:

- Practice is needed to arrange the original song because the manipulation speed required is too fast for arrangement at sight.
- The system should be unified into one device such as a tablet because it is strange that the screen and the keyboard are separated.

The difficulty on playing at sight is also seen in an ordinary piano performance.

The following opinions were obtained on the signal processing:

- Pitch modification sometimes significantly degrades the sound quality.
- The gap between the correct pitch and the estimated pitch makes me feel unpleasant when the estimated notes were shown on the screen.

Considering these opinions, we assume that the former problem occurs when the F0s of the singing voices contain some vibratos. PSOLA algorithm do not work well in such situations because it assumes that the F0s are invariant within one note. On the other hand, the latter problem occurs when extracted F0s contain some accompaniments and therefore some unnatural notes are generated.

5.3 Discussion

The opinions obtained encourage us to tackle following three problems: (1) making it easier to master the system, (2) improving sound quality after pitch modification, and (3) adding more functions. First, we plan to make the system easier to use by implementing it on a tablet that has a touchscreen as an input device to unify the device that displays information with the device that a user operates. By unifying the device, a user is free from paying attention to two different devices (*i.e.* the screen and the keyboard) at once and able to concentrate on their hand. Second, we try to use short-time Fourier transform and handling the signal in the time-frequency domain (*e.g.*, a phase vocoder). The sound quality after pitch modification is low due to the vibratos in singing voices. In the time-frequency domain, on the other hand, the envelope of the voice spectrum can be estimated and therefore it may contribute to preserving the individuality of the original singer. This improvement may make the arranged voice clear enough for a listener to recognize the original lyrics. Third, we are considering the feasibility of adding functions to change the volume and to assign two or more notes at the same time, which were desired in the subjective experiment.

6. CONCLUSION

This paper proposed a system for singing voice arrangement in real time. The subjective experimental results indicated that the proposed system provided users with the experience that they were arranging a vocal part as if playing an instrument. On the other hand, we found that the system is hard to use at first because the screen and the keyboard are separate devices. We therefore plan to implement this system in tablet that has a touchscreen as an input device. We also plan to improve the sound quality by using other signal processing methods such as a phase vocoder. Moreover, we plan to extend the function for arrangement of an accompaniment part and a drum part as well as a vocal part. This extension would provide a new user experience, the rearrangement of the component parts in songs. While this kind of arrangement is already partially realized by electronic organs, our goal is different from them because we aim to use the original parts contained in the original song.

Acknowledgments

This study was partially supported by JST CREST Project, JST ACCEL Project, JSPS KAKENHI 24220006, 26700020, 26280089, 16H01744. In this study, We used RWC music database (popular).

7. REFERENCES

[1] K. Yoshii *et al.*, “Drumix: An audio player with real-time drum-part rearrangement functions for active mu-

sic listening," *Trans. of IPSJ*, vol. 48, no. 3, pp. 1229–1239, 2007.

[2] N. Yasuraoka *et al.*, "Changing timbre and phrase in existing musical performances as you like: manipulations of single part using harmonic and inharmonic models," in *Proc. of the 17th ACM Int. Conf. on Multimedia*, 2009, pp. 203–212.

[3] H. Kenmochi *et al.*, "Vocaloid–commercial singing synthesizer based on sample concatenation," in *Proc. of the Interspeech Conf.*, 2007, pp. 4009–4010.

[4] Y. Ikemiya *et al.*, "Singing voice analysis and editing based on mutually dependent f0 estimation and source separation," in *Proc. of ICASSP*, 2015, pp. 574–578.

[5] R. Nishikimi *et al.*, "Musical note estimation for F0 trajectories of singing voices based on a Bayesian semi-beat-synchronous HMM," in *Proc. of ISMIR*, 2016, pp. 461–467.

[6] M. Goto *et al.*, "Songle: A web service for active music listening improved by user contributions," in *Proc. of ISMIR*, 2011, pp. 311–316.

[7] M. Mauch *et al.*, "Song prompter: An accompaniment system based on the automatic alignment of lyrics and chords to audio," in *Late-breaking session at the 10th ISMIR*, 2010.

[8] T. M. Nakra *et al.*, "The ubs virtual maestro: an interactive conducting system," in *NIME*, 2009, pp. 250–255.

[9] I. Simon *et al.*, "Mysong: automatic accompaniment generation for vocal melodies," in *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, 2008, pp. 725–734.

[10] M. McVicar *et al.*, "Autorhythmguitar: Computer-aided composition for rhythm guitar in the tab space," in *Proc. of Joint ICMC and SMC Conf.*, 2014.

[11] L. Yi *et al.*, "Automatic generation of four-part harmony," in *Proc. of UAI Applications Workshop*, 2007.

[12] R. B. Dannenberg *et al.*, "Human-computer music performance: From synchronized accompaniment to musical partner," in *Proc. of SMC*, 2013.

[13] R. B. Dannenberg, "An on-line algorithm for real-time accompaniment," in *ICMC*, 1984, pp. 193–198.

[14] A. Cont, "A coupled duration-focused architecture for real-time music-to-score alignment," *IEEE Trans. on PAMI*, vol. 32, no. 6, pp. 974–987, 2010.

[15] K. Yamamoto *et al.*, "Livo: Sing a song with a vowel keyboard," *JIP*, vol. 24, no. 3, pp. 460–468, 2016.

[16] K. Tsuzuki *et al.*, "Unisoner: An interactive interface for derivative chorus creation from various singing voices on the web," in *Proc. of ICMC*, 2014, pp. 790–797.

[17] Z. Rafii *et al.*, "Music/voice separation using the similarity matrix," in *Proc. of ISMIR*, 2012, pp. 583–588.

[18] P.-S. Huang *et al.*, "Singing-voice separation from monaural recordings using robust principal component analysis," in *Proc. of ICASSP*, 2012, pp. 57–60.

[19] A. Laaksonen, "Automatic melody transcription based on chord transcription," in *Proc. of ISMIR*, 2014, pp. 119–124.

[20] M. P. Ryynänen *et al.*, "Automatic transcription of melody, bass line, and chords in polyphonic music," *Computer Music Journal*, vol. 32, no. 3, pp. 72–86, 2008.

[21] Z. Udo *et al.*, *DAFX - Digital Audio Effects*. Wiley, 2002.

[22] S. Roucos *et al.*, "High quality time-scale modification for speech," in *Proc. of ICASSP*, 1985, pp. 493–496.

[23] C. Hamon *et al.*, "A diphone synthesis system based on time-domain prosodic modifications of speech," in *Proc. of ICASSP*, 1989, pp. 238–241.

[24] M. Goto *et al.*, "RWC music database: popular, classic, and jazz music databases," in *Proc. of ISMIR*, 2002, pp. 287–288.

Evaluation of the Learnability and Playability of Pitch Layouts in New Musical Instruments

Jennifer MacRitchie and Andrew J. Milne
The MARCS Institute for Brain, Behaviour and Development
Western Sydney University
j.macritchie@westernsydney.edu.au a.milne@westernsydney.edu.au

ABSTRACT

Certain properties of isomorphic layouts are proposed to offer benefits to learning and performances on a new musical instrument. However, there is little empirical investigation of the effects of these properties. This paper details an experiment that examines the effect of pitch adjacency and shear on the performances of simple melodies by 24 musically-trained participants after a short training period. In the adjacent layouts, pitches a major second apart are adjacent. In the unsheared layouts, major seconds are horizontally aligned but the pitch axis is slanted; in the sheared layouts, the pitch axis is vertical but major seconds are slanted. Qualitative user evaluations of each layout are collected post-experiment. Preliminary results are outlined in this paper, focusing on the themes of learnability and playability. Users show strong preferences towards layouts with adjacent major seconds, focusing on the potential for learning new pitch patterns. Users confirm advantages of both unsheared and sheared layouts, one in terms of similarity to traditional instrument settings, and the other to ergonomic benefits. A model of participants' performance accuracy shows that sheared layouts are learned significantly faster. Results from this study will inform new music instrument/interface design in terms of features that increase user accessibility.

1. INTRODUCTION

Designers of new or extended musical instruments are often concerned with ensuring accessibility for users either with no previous musical experience, or for those who already have training in another instrument, so that they can easily alter/learn new techniques. Several claims regarding the optimal pitch layout of new electronic instruments/interfaces have been made, but as yet there is little empirical investigation of the factors that may enhance or disturb learning and performance on these devices.

1.1 Isomorphic Layout Properties

Since the nineteenth century, numerous music theorists and instrument builders have conjectured that *isomorphic pitch layouts* provide important advantages over the conventional pitch layouts of traditional musical instruments [1–4]. Indeed, a number of new musical interfaces have used isomorphic layouts (e.g., Array Mbira [5], Thummer [6], Soundplane [7], AXiS-49 [8], Musix Pro [9], LinnStrument [10], Lightpad Block [11], Terspstra [12]).

An isomorphic layout is one where the spatial arrangement of any set of pitches (a chord, a scale, a melody, or a complete piece) is invariant with respect to musical transposition. This contrasts with conventional pitch layouts on traditional musical instruments; for example, on the piano keyboard, playing a given chord or melody in a different transposition (e.g., in a different key) typically requires changing fingering to negotiate the differing combinations of vertically offset black and white keys.

Isomorphic layouts also have elegant properties for microtonal scales, which contain pitches and intervals "between the cracks" of the piano keyboard [13]. Although strict twelve-tone equal temperament (12-TET) is almost ubiquitous in contemporary Western music, different tunings are found in historical Western and in non-Western traditions. Isomorphic layouts may, therefore, facilitate the performance of music both within and beyond conventional contemporary Western traditions.

In this paper, we do not compare isomorphic and non-isomorphic layouts. Instead, we focus on how different isomorphic layouts impact on learnability and playability. This is because there are an infinite number of unique isomorphic layouts: they all share the property of transpositional invariance (by definition) but they differ in a number of other ways that may plausibly impact their usability. For example, successive scale pitches, like C, D, and E, are spatially adjacent in some isomorphic layouts while in others they are not; additionally, in some isomorphic layouts, pitches are perfectly correlated to a horizontal or vertical axis while in others they are not [14]. In some layouts, octaves may be vertically or horizontally aligned; in others, they are slanted. [1]

Properties such as these are typically non-independent: improving one (e.g., pitch axis orientation) may worsen another (e.g., octave axis orientation). Choosing an optimal layout thus becomes a non-trivial task that requires knowledge of the relative importance of the different properties.

To shed light on this, the experiment presented in this paper explores how two independent spatial transformations

[1] With respect to the instrumentalist, the "horizontal" axis runs from left to right, the "vertical" axis from bottom to top or from near to far.

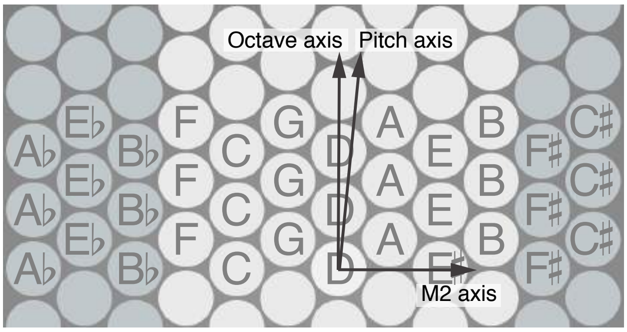


Figure 1. Layout $A'S'$.

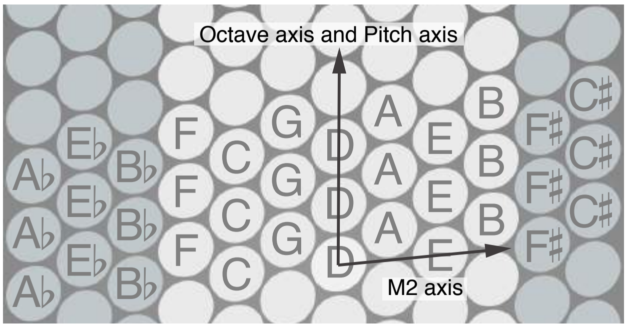


Figure 2. Layout $A'S$.

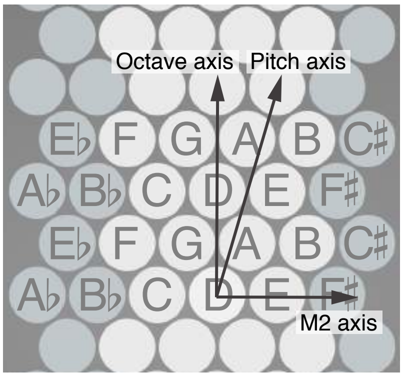


Figure 3. Layout AS' (the Wicki layout [3]).

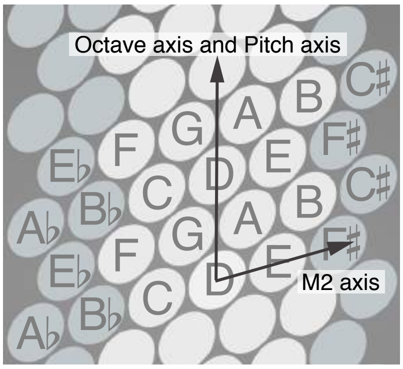


Figure 4. Layout AS.

of isomorphic layouts – adjacency and shear – impact on their learnability and playability for melodies. The four resulting layouts are illustrated in Figures 1–4. Each figure shows how pitches are positioned, and the orientation of three axes that we hypothesize will impact on the layout's usability. Each label indicates whether the layout has adjacent major seconds or not (A and A', respectively) and whether it is sheared or not (S and S', respectively). The three axes are the *pitch axis*, the *octave axis*, and the *major second axis*, as now defined (the implications of these three axes, and why they may be important, are detailed in 1.1.2).

- The *pitch axis* is any axis onto which the orthogonal (perpendicular) projections of all button centres are proportional to their pitch; for any given isomorphic layout, all such axes are parallel [15].
- The *octave axis* is here defined as any axis that passes through the closest button centres that are an octave apart.
- The *major second axis* (*M2 axis*, for short) is here defined as any axis that passes through the closest button centres that are a major second apart. [2]

1.1.1 Adjacent (A) or Non-Adjacent (A') Seconds

Scale steps (i.e., major and minor seconds) are, across cultures, the commonest intervals in melodies [16]. It makes sense for such musically privileged intervals also to be spatially privileged. An obvious way of spatially privileging intervals is to make their pitches adjacent: this makes transitioning between them physically easy, and makes them visually salient. However, when considering bass or harmony parts, scale steps may play a less important role. This suggests that differing layouts might be optimal for differing musical uses.

The focus of this experiment is on melody so, for any given layout, we tested one version where all major seconds are adjacent and an adapted version where they are nonadjacent (minor seconds were non-adjacent in both versions). Both types of layouts have been used in new musical interfaces; for example, the Thummer (which used the Wicki layout (Fig. 3)) had adjacent major seconds, while the AXiS-49 (which uses a Tonnetz-like layout) [17]) has non-adjacent seconds but adjacent thirds and fifths.

1.1.2 Sheared (S) or Unsheared (S')

We conjecture that having any of the above-mentioned axes (pitch, octave, and M2) perfectly horizontal or perfectly vertical makes the layout more comprehensible: if the pitch axis is vertical or horizontal (rather than slanted), it allows for the pitch of buttons to be more easily estimated by sight, thereby enhancing processing fluency. Similar advantages hold for the octave and M2 axes: scales typically repeat at the octave, while the major second is the commonest scale step in both the diatonic and pentatonic scales that form the backbone of most Western music.

However, changing the angle of one of these axes typically requires changing the angle of one or both of the

[2] When considering tunings different to 12-TET (e.g., meantone or Pythagorean), alternative – but more complex – definitions for the octave and M2 axes become useful.

others, so their independent effects can be hard to disambiguate.

A way to gain partial independence of axis angles is to shear the layout parallel with one of the axes – the angle of the parallel-to-shear axis will not change while the angles of the other two will.[3] As shown by comparing Figure 1 with Figure 2, or by comparing Figure 3 with Figure 4, we used a shear parallel with the octave axis to create two versions of the non-adjacent layout and two versions of the adjacent layout: each unsheared version ($A'S'$ or AS') has a perfectly horizontal M2 axis but a slanted (non-vertical) pitch axis; each sheared version version ($A'S$ or AS) has a slanted (non-horizontal) M2 axis but a vertical pitch axis. In both cases the octave axis was vertical.

In this investigation, therefore, we remove any possible impact of the octave axis orientation; we cannot, however, quantitatively disambiguate between the effects of the pitch axis and the M2 axis, although qualitative information can provide some insight into their independent effects (see Sec. 3.1).

Sheared and unsheared layouts are found in new musical interfaces – the Thummer and AXiS-49 both have unsheared layouts; the crowdfunded Terpstra keyboard uses a sheared layout.

1.2 Motor Skill Acquisition

Learning a new musical instrument requires a number gross and fine motor skills (in order to physically play a note), and sensory processing (of feedback from the body and of auditory features, e.g., melody, rhythm, timbre) [18]. For trained musicians adapting to a new instrument, they may be required to learn a new pitch mapping, which means they must adjust their previously learned motor-pitch associations [19]. In learning a motor skill there are three general stages [20]:

- a *cognitive stage*, encompassing the processing of information and detecting patterns. Here, various motor solutions are tried out, and the performer finds out which solutions are most effective.

- a *fixation stage*, when the general motor solution has been selected, and a period commences where the patterns of movement are perfected. This stage can last months, or even years.

- an *autonomous stage*, where the movement patterns do not require as much conscious attention on the part of the performer.

Relating this to how a performer learns to play a particular a musical instrument, we define the first cognitive stage as *learnability* and the second fixation stage as *playability*. Essentially, learning the motor-pitch associations of a new instrument requires the performer to perceive and remember pitch patterns. Once these pitch patterns are learned, the performer becomes more focused on eliminating various sources of motor error.[4] To test how well the participants have learned the new layouts and perfected their motor pattern, we are interested in the transfer of learning from one task to another. For instance, a piano player will practice scales, not only so they will achieve a good performance of scales, but so they can play scale-like passages in other musical pieces well. In our study, we designed a training and testing paradigm for the different pitch layouts such that the test involved a previously un-practised, but familiar (in pitch) melody.

1.3 Study Design

For this experiment, we are interested in examining how features of a pitch layout affect the ease with which musicians can adapt to it. Musically experienced participants played three different layouts successively. In each such layout, they received an equivalent training program and then performed a test melody four times. The independent variables were *Adjacency* $\in \{0,1\}$, *Shear* $\in \{0,1\}$, *LayoutNo* $\in \{0,1,2\}$ (respectively the first, second, or third layout played), and *PerfNo* $\in \{0,1,2,3\}$ (respectively the first, second, third, or fourth performance of the melody in each layout). Participants were presented the layouts in different orders to permit analysis of ordering effects. In using musically-experienced participants, we assume that they share a high level of musical information processing skill, thereby reducing its impact on the results. Participants' preferences were elicited in a semi-structured interview, and the accuracy of their performances was numerically assessed from their recorded MIDI data.

2. METHODS

2.1 Participants

24 participants were recruited (age mean = 26, range: 18-44) with at least 5 years of musical experience on any one instrument (excluding the voice).

2.2 Materials

2.2.1 Hardware and Software

The software sequencer Hex [14] was modified to function as a multitouch MIDI controller, and presented on an Asus touch screen notebook, as shown in Figure 5. Note names are not shown on the interface, but middle D is indicated with a subtly brighter button to serve as a global reference. The position of middle C was indicated to participants, this being the starting pitch of every scale, arpeggio, or melody they played.

In order to present training sequences effectively, both aurally and visually, Hex's virtual buttons were highlighted in time with a MIDI sequence. All training sequences were at 90 bpm and introduced by a two-bar metronome count.

[3] A *shear* is a spatial transformation in which points are shifted parallel to an axis by a distance proportional to their distance from that axis. For example, shearing a rectangle parallel to an axis running straight down its middle produces a parallelogram; the sides that are parallel to the shear axis remain parallel to it, while the other two sides rotate.

[4] Because achieving motor autonomy is a lengthy process – one that can seldom be captured by experiments – our current study focuses on only the first two elements of motor learning.

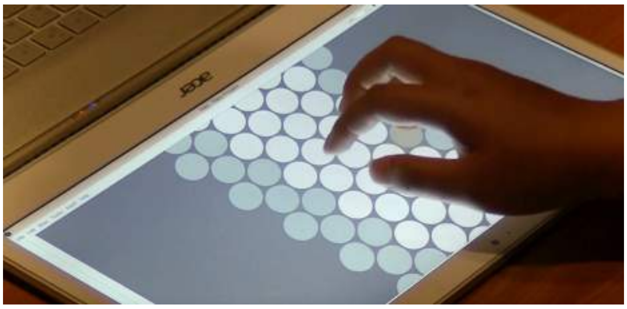


Figure 5. The multitouch interface used in the experiment.

2.2.2 *Musical Tasks*

Melodies for musical tasks were chosen to be single-line sequences to be performed solely with the right hand. The training melodies consisted of a set of C major scales and a set of arpeggios (again only using the notes of the C major scale) spanning two octaves, and all starting and ending on middle C. The well-known nursery rhyme *Frère Jacques* was used as a test melody; it too was played in C major and began and ended on middle C.

2.3 Procedure

Participants played three out of the four layouts under consideration: all 24 participants played both AS' and AS, with 12 participants each playing either $A'S$ or $A'S'$.

The layouts were presented in four different sequences, with each sequence played by 6 participants: AS' then $A'S'$ then AS; or AS' then $A'S$ then AS; or AS then $A'S'$ then AS'; or AS then $A'S$ then AS'. This means that the non-adjacent seconds layouts ($A'S'$ and $A'S$) were always presented second, and that participants who started with the unsheared adjacent layout (AS') finished with the sheared adjacent layout (AS), and vice versa.

2.3.1 *Training Paradigm*

For each of their three layouts, participants were directed through a 15-minute training paradigm involving i) scales and ii) arpeggios. For each stage, this involved:

1. watching the sequence once as demonstrated by audiovisual highlighting
2. playing along with the audiovisual highlighted sequence three times
3. reproducing the sequence in the absence of audiovisual highlighting, for four consecutive performances.

All demonstration sequences and participant performances were played in time with a 90bpm metronome, and recorded as MIDI files.

2.3.2 *Testing*

A final production task asked participants to a well-known nursery rhyme – *Frère Jacques*. Participants first heard an audio recording of the nursery rhyme to confirm their knowledge of the melody. They were then given 20 seconds initially to explore the layout and find the correct notes before giving four consecutive performances. Again, these performances were instructed to be played in time with a 90 bpm metronome. Although this represents a fairly simple task, the nursery rhyme was chosen as it facilitated measurement of participants' skill with each particular layout. We assume that as the participants' memory for the melody was intact, their performance would only be affected by their memory of the layout itself.

2.3.3 *Post-Experiment Interview*

In a semi-structured interview taking place after the training paradigm and test performances concluded, participants were asked to choose their preferred layout from the three presented. Participants were asked to detail what they liked or disliked specifically about their preferred layout (and any of the others). When required, the experimenter would prompt participants to consider how it felt playing the scales, arpeggios, or melodies from the training section.

2.4 Data Analysis

2.4.1 *Qualitative User-Evaluation Statements*

Preferences for any specific layout were noted only for participants who clearly indicated a distinct preference (23 out of 24). 102 statements were made by the 24 participants regarding their likes and dislikes of the layouts they were exposed to. Eight statements were categorized as preferences for a specific layout feature without any clarification. Two statements were deleted as they neither identified a preference nor discussed any particular layout feature. The remaining 92 statements were coded independently by the two authors, categorized as issues of *Learnability* or *Playability*. Mention of visual aspects of a layout, or any reference to a collection of buttons in rows or patterns that may represent cognitive processing on the part of the performer was classified as *Learnability* (see Section 1.2). Mention of any physical aspect of playing that may represent the perfection of motor skills was classified as *Playability*. Each category was further classified into whether statements made were positive or negative.

2.4.2 *Quantitative MIDI Performance Data*

Each MIDI performance was analyzed for accuracy of pitches by comparing against an ideal performance of the melody at 90bpm. To account for participants who may not have been able to play the full melody, a score was given for the number of 'correct' notes played. This score was adjusted by giving negative points for any wrong or extra notes. The maximum score for a performance in this case was 32.

	Learnability		Playability		Learnability/Playability	
	Positive	Negative	Positive	Negative	Positive	Negative
A	25	3	13	2	8	1
A'	3	24	2	10	0	0
Both	1	0	0	0	0	0
AS	6	3	7	0	5	1
AS'	9	0	6	2	2	0
Both	10	0	0	0	1	0
$A'S$	1	15	1	6	0	0
$A'S'$	2	9	1	4	0	0

Table 1. User evaluation statements categorized by Learnability, Playability, or Learnability/Playability. These statements are separated in terms of the positive or negative meaning, and the particular layout they were referring to in terms of *Adjacency* (A, A') and then *Shear* (S, S').

3. RESULTS

3.1 User Evaluations

3.1.1 User Preferences

There was a strong consensus amongst the participants in terms of *Adjacency* (22 participants selecting either layout AS or AS', 1 participant selecting layout AS). However, no conclusive preference was registered in terms of *Shear*, with 12 participants selecting layout AS, and 10 participants selecting layout AS'.

3.1.2 User Statements

Participants made from 0 to 7 qualitative statements each (mode = 4). Only two participants made no useable statements. Table 1 shows the statements as they referred to at first the *Adjacency* of the layout, and then as they refer to the *Shear*.

3.1.3 Learnability

The large majority of statements made (61%) concerned the learnability of a certain layout. 29 statements were positive, 25 of which related to the adjacent M2 layouts (AS and AS'). Performers noted that the pattern in the adjacent M2 layout was easy to grasp, easier to remember and see where the notes were, implying that they quickly perceived the appropriate pitch patterns for the scale and arpeggio exercises. The consecutive adjacent notes were mentioned explicitly as being helpful to remembering where pitches were, as well as the “3-4 pattern” formed by the diatonic scale. The three positive statements made in relation to non-adjacent M2 layouts all concerned the adjacent octave notes. One statement again referred to the aligned octaves in reference to both adjacency types. 24 negative statements were made regarding the layout adjacency, particularly in nonadjacent (A') layouts. These referred to the difficulty in “seeing the lines”, “finding the notes” and it being harder to determine which note was which. Shear in these non-adjacent layouts did not appear to particularly help or worsen the confusion. Negative statements that referred specifically to the shear of the adjacent layouts referred to layout AS. Three statements from two participants clarified that the shear contributed to confusion in being able to separate out the different rows of buttons.

3.1.4 Playability

29% of statements referred to the playability of the layouts. Again, the majority of positive statements referred to the adjacent layouts (AS and AS') and the majority of negative statements referred to the non-adjacent layouts ($A'S$ and $A'S'$). Five participants made six positive statements in total regarding the unsheared layout AS', mentioning that the straight line in particular helped them to play, and that the circles made the notes spaced out a bit more than the ovals (an effect of the shear). They also liked fewer notes to a row (in comparison the the non-adjacent layouts). The statements on playability for the sheared layout often contained specific details to do with the shear itself. Participants liked the ergonomics of the slant, commenting on its versatility, the better angle for the hand, and the ease of moving between rows. Two positive statements referring to the non-adjacent layouts (one each for $A'S$ and $A'S'$), commented that the larger width between adjacent notes made it easier to play, and that the layout gave them more freedom. The negative statements (largely referring to non-adjacent layouts), commented on having to physically spread the fingers out more to play consecutive notes. The close proximity of non-adjacent notes (due to the interleaved layout of buttons), also contributed to some discomfort as performers mentioned that they often hit the wrong note on the wrong “row”. Two statements referred to the unsheared adjacent M2 layout (AS'), commenting that the vertical alignment of rows made the performers more likely to miss a note as their hand was in the way.

3.1.5 Other

The remaining 10% of statements did not fit into one category as they mentioned both cognitive and physical aspects of playing. These all referred to the adjacent layouts. Two participants stated that AS' was the easiest layout to “get used to”, particularly as it was very similar to other instruments like the piano. The statements referring to the

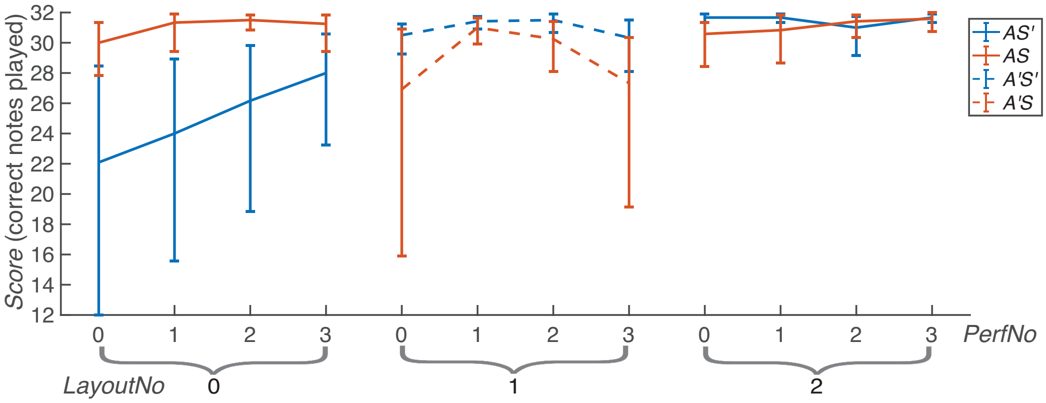


Figure 6. Performance scores, averaged across participants, as a function of layout number (where 0 codes the first layout presented to the participant, 1 codes their second layout, 2 codes their final layout), and performance number (where 0 codes their first performance in each layout, 1 codes their second performance, etc.). The error bars show 95% confidence intervals obtained by bootstrapping.

Fixed Effects	Estimate	SE	t-Stat	DF	p-Value	95% CI
(*Intercept*)	26.25	1.49	17.65	274	.000	$[23.32, 29.17]$
Adjacency	−3.38	1.28	−2.65	274	.008	$[-5.90, -0.87]$
LayoutNo	3.95	0.81	4.86	274	.000	$[2.35, 5.55]$
Adjacency×*Shear*	4.70	1.85	2.54	274	.012	$[1.06, 8.33]$
Shear×*LayoutNo*	−3.05	1.25	−2.44	274	.015	$[-5.51, -0.59]$
PerfNo×*LayoutNo*	−1.04	0.39	−2.69	274	.008	$[-1.80, -0.28]$

Table 2. A linear mixed effects model of *Score* with random effects on the intercept grouped by participant. For simplicity, non-significant predictors are not shown, although they were included in the model.

sheared, pitch-adjacent layout AS commented on the angle of the rows making it easier to visualize where the fingers would go next and an intuitive sense of moving upwards. As these statements hadn't specifically stated whether this related to learning/visualizing the patterns of pitches or the actual ergonomics of playing, we did not further categorize them.

There were also eight extra statements (separate to those in Table 1) that noted a preference for a specific feature of one or more of the layouts. Participants here noted a preference for the circle rather than oval buttons (an effect of the shear), and a preference for the "compact" layout that minimized gaps between notes (adjacency). Two statements noted the flat (unsheared) layouts were easier, and one noted that the sheared, adjacent layout (AS) would work well as a split screen when considering two-handed playing.

3.1.6 Summary

Participants clearly show a preference in terms of adjacent M2 layouts. Their main concerns are reflected in the learnability of a layout, and being able to figure out the pitch patterns. Playability is concerned particularly in statements regarding the shear of a layout.

3.2 MIDI Data

The scores (correct notes played) for performances of *Frère Jacques*, averaged across participants, are summarized in Figure 6 as a function of layout type (AS, AS', $A'S$, or $A'S'$); layout number (where 0 is the first layout presented to the participant, 1 is the second layout presented, 2 is the final layout); performance number (where 0 is the first performance in any given layout, 1 is the second performance, etc.). The 95% confidence intervals were obtained with 10,000 bootstrap samples.

The correlation between participants' preferences and their correct note score was $r = -.05$ and not significant ($p = .441$).

3.2.1 Model 1

To analyse which factors, and which of their interactions, impact on participants' scores, a linear mixed effects model was run with *Score* as the dependent variable, and the following fixed effects predictors: *Intercept*, *Adjacency*, *Shear*, *PerfNo*, *LayoutNo*, *Adjacency*×*Shear*, *Adjacency*×*PerfNo*, *Shear*×*PerfNo*, *Shear*×*LayoutNo*, *PerfNo*×*LayoutNo*, *Adjacency*×*Shear*×*PerfNo*, *Shear*×*PerfNo*×*LayoutNo*. Because all non-adjacent layouts were second, interactions between *Adjacency* and *LayoutNo* are not linearly independent of the main effects and so cannot also be included in the model. The intercept was treated as a random effect grouped by participant, which allows the model to take account of participants' differing aptitudes regarding the task as a whole.

The strongest significant effect is the interaction *Adjacency*×*Shear*, which indicates the generally superior per-

Fixed Effects	Estimate	SE	t-Stat	DF	p-Value	95% CI
(*Intercept*)	21.72	2.03	10.70	91	$.000$	$[17.69, 25.75]$
Shear	8.71	2.86	3.05	91	$.003$	$[3.21, 13.50]$
PerfNo	2.14	0.54	3.99	91	$.000$	$[1.08, 3.21]$
Shear×*PerfNo*	-1.75	0.75	-2.33	91	$.022$	$[-3.25, -0.26]$

Table 3. A linear mixed effects model of the scores obtained during the first presented layout, which was either AS' or AS.

formance of the sheared and adjacent layout *AS*. The significant negative conditional effect of *Adjacency* indicates that *Adjacency* reduces the score with unsheared layouts. This results from the fact that non-adjacent layouts were only presented second, by which time learning had already resulted in *Score* approaching the ceiling. This can be seen in Figure 6 by observing the course of the blue lines (the unsheared layouts) – the initial score is relatively low, but rapidly improves due to learning and has hit the ceiling by the time the second (non-adjacent) layout is presented.

The conditional effect of *LayoutNo* indicates that it has a strong positive impact across the first performance of unsheared layouts; the negatively-weighted interaction between *PerfNo* and *LayoutNo* shows that the impact of *LayoutNo* reduces as *PerfNo* increases, and vice versa.

The negatively-weighted interaction between *LayoutNo* and *Shear* shows that the positive impact of *Shear* is essentially swamped by the time the final layout was presented – another consequence of the ceiling effect.

3.2.2 Model 2

The ceiling effect, established above, suggests that the results that are most indicative of the initial learning of a new layout are those that arise during the first presented layout (i.e., when *LayoutNo* $= 0$). Table 3 summarizes a regression of the scores obtained during only the first presentation on *PerfNo* and *Shear* and their interaction, with a random effect on the intercept grouped by participant.

This model shows that – for the first presented layout, which always had adjacent seconds – both *Shear* and *PerfNo* have strong positive effects on *Score*. However, the interaction shows that the positive impact of *PerfNo* is much smaller when the layout is sheared, and that the positive impact of *Shear* is somewhat reduced as *PerfNo* increases. As before, these inferences can be readily identified in Figure 6.

3.3 Limitations

The current analysis considers the correct note score for performances of a well-known melody, where the maximum score is 32. We see a strong ceiling effect, particularly for layout AS where performances are close to the maximum number of correct notes possible. The musical task used for testing is simple to ensure that performance is affected only by the layout manipulations and not by participants' ability to memorise a new melody. What may be more revealing is an analysis of the timing with which participants are able to play these melodies.

The design of the experiment may also have contributed to the preferences and disparity of statements concerning *Adjacency*. Each participant was exposed to three different layouts: two with adjacent major seconds (sheared and unsheared); one with non-adjacent major seconds (sheared or unsheared). The adjacent M2 layout always came second; a more complete set of orderings would be desirable. The experiment tests for right-handed performance of single-note melodies after a short training paradigm.

Results presented on a transfer task immediately after training show an effect of shear on learning of this single-note melody, however, as different tasks are theorised to benefit from different layouts, effects may differ for performance of chords or left-handed bass lines, not to mention more complicated tasks such as modulation or improvisation. Further testing would also be necessary to determine the long-term effects from training.

4. CONCLUSIONS

Non-standard pitch layouts have been proposed since the late nineteenth century and, over just the last ten years, a number of products utilizing novel pitch layouts have been demonstrated at conferences such as SMC and NIME [6,7,9,21–23], or released commercially (as previously referenced). Purported advantages of such layouts are often theoretically plausible but they are rarely subjected to formal experimental tests. There is a multiplicity of different factors that may play a role in the learnability and playability of pitch layouts; furthermore, these factors are often non-independent. We have used two independent spatial manipulations (*Adjacency* and *Shear*) – realized with a multitouch interface – to tease apart some of the underlying factors that affect learnability and playability. These spatial manipulations allow us to gain insight into the impact of: spatially privileging major seconds by adjacency or angle; spatially privileging the pitch axis by angle.

The quantitative MIDI analysis indicates that shear provides a significant improvement in participants' ability to learn quickly and to play accurately – indeed, even on first performance, their correct note score was typically very good. Having the non-adjacent M2 layouts presented only second makes the impact of this factor on learnability harder to assess. Fortunately the user evaluations provide insight here.

The user evaluations suggest that M2 adjacency has a strong effect on the learnability of a layout, with adjacent M2 layouts easier to grasp and process, perhaps due to familiarity with previous motor-pitch associations, and perhaps due to their importance as an interval in Western music. In the user evaluations, shear is only considered as a secondary concern, when a motor-pitch pattern has already been acquired and needs to be perfected.

The quantitative analysis does not allow us to determine whether the key factor behind shear's impact is how it changes the direction of the pitch axis or how it changes the direction of the M2 axis. The user evaluations, however, suggest that both underlying factors play a role. Future research may aim to differentiate between these two explanations, and to collect more data to eliminate the ordering effect on adjacency.

Acknowledgments

Anthony Prechtl, who coded the original version of Hex and who specially adapted it for this project to make it receptive to multitouch.

Dr Andrew Milne is the recipient of an Australian Research Council Discovery Early Career Award (project number DE170100353) funded by the Australian Government.

5. REFERENCES

[1] R. H. M. Bosanquet, *Elementary Treatise on Musical Intervals and Temperament*. London: Macmillan, 1877.

[2] H. L. F. Helmholtz, *On the Sensations of Tone as a Physiological Basis for the Theory of Music*. New York: Dover, 1877.

[3] K. Wicki, "Tastatur für musikinstrumente," Neumatt (Münster, Luzern, Schweiz), Oct. 1896.

[4] D. Keislar, "History and principles of microtonal keyboards," *Computer Music Journal*, vol. 11, no. 1, pp. 18–28, 1987. [Online]. Available: http://ccrma.stanford.edu/STANM/stanms/stanm45

[5] Array Instruments, "The Array Mbira," 2017. [Online]. Available: http://www.arraymbira.com

[6] G. Paine, I. Stevenson, and A. Pearce, "Thummer mapping project (thump) report," in *Proceedings of the 2007 International Conference on New Interfaces for Musical Expression (NIME07)*, L. Crawford, Ed., June 2007, pp. 70–77.

[7] R. Jones, P. Driessen, A. Schloss, and G. Tzanetakis, "A force-sensitive surface for intimate control," in *Proceedings of the 2009 International Conference on New Interfaces for Musical Expression (NIME09)*, 2009, pp. 236–241.

[8] C-Thru Music, "The AXiS-49 USB music interface," 2017. [Online]. Available: http://www.c-thru-music.com/cgi/?page=prod_axis-49

[9] B. Park and D. Gerhard, "Rainboard and Musix: Building dynamic isomorphic interfaces," in *Proceedings of the International Conference on New Interfaces for Musical Expression*. Daejeon, Republic of Korea, 2013, pp. 319–324.

[10] Roger Linn Design, "LinnStrument: A revolutionary expressive musical performance controller," 2017. [Online]. Available: http://www.rogerlinndesign.com/linnstrument.html

[11] ROLI, "ROLI : BLOCKS," 2017. [Online]. Available: https://roli.com/products/blocks

[12] S. Terpstra and D. Horvath. (2017) Terpstra keyboard. [Online]. Available: http://terpstrakeyboard.com

[13] A. J. Milne, W. A. Sethares, and J. Plamondon, "Isomorphic controllers and Dynamic Tuning: Invariant fingering over a tuning continuum," *Computer Music Journal*, vol. 31, no. 4, pp. 15–32, Dec. 2007.

[14] A. Prechtl, A. J. Milne, S. Holland, R. Laney, and D. B. Sharp, "A MIDI sequencer that widens access to the compositional possibilities of novel tunings," *Computer Music Journal*, vol. 36, no. 1, pp. 42–54, 2012.

[15] A. J. Milne, W. A. Sethares, and J. Plamondon, "Tuning continua and keyboard layouts," *Journal of Mathematics and Music*, vol. 2, no. 1, pp. 1–19, 2008.

[16] P. G. Vos and J. M. Troost, "Ascending and descending melodic intervals: Statistical findings and their perceptual relevance," *Music Perception*, vol. 6, no. 4, pp. 383–396, 1989.

[17] L. Euler, *Tentamen novae theoriae musicae ex certissismis harmoniae principiis dilucide expositae*. Saint Petersburg: Saint Petersburg Academy, 1739.

[18] R. J. Zatorre, J. L. Chen, and V. B. Penhune, "When the brain plays music : auditory – motor interactions in music perception and production," *Nature Reviews: Neuroscience*, 2007.

[19] P. Q. Pfordresher, "Musical training and the role of auditory feedback during performance," *ANNALS OF THE NEW YORK ACADEMY OF SCIENCES*, vol. 1252, pp. 171–178, 2012.

[20] R. A. Schmidt and T. D. Lee, *Motor Control and Learning, 5th Ed.* Champaign, IL: Human Kinetics, 2011.

[21] A. J. Milne, A. Xambó, R. Laney, D. B. Sharp, A. Prechtl, and S. Holland, "Hex player—a virtual musical controller," in *Proceedings of the 2011 International Conference on New Interfaces for Musical Expression (NIME11)*, A. R. Jensenius and R. I. Godøy, Eds., Oslo, Norway, 2011, pp. 244–247.

[22] L. Bigo, J. Garcia, A. Spicher, and W. E. Mackay, "PaperTonnetz: Music composition with interactive paper," in *Proceedings of the 9th Sound and Music Computing Conference*, Copenhagen, Denmark, 2012, pp. 219–225.

[23] T. W. Hedges and A. P. McPherson, "3D gestural interaction with harmonic pitch space," in *Proceedings of the Sound and Music Computing Conference 2013*, Stockholm, Sweden, 2013, pp. 103–108.