



SBCM 2017

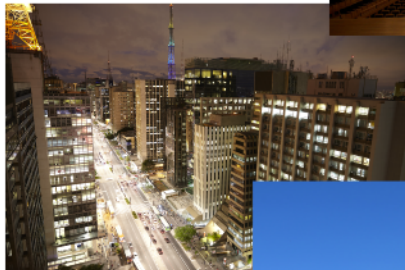
# PROCEEDINGS

16th Brazilian Symposium  
on Computer Music

[compmus.ime.usp.br/sbcm/2017](http://compmus.ime.usp.br/sbcm/2017)

University of São Paulo

September 3-6



Ficha catalográfica elaborada pelo Setor de Processamento Técnico da Divisão de Biblioteca da UFSJ

Brazilian Symposium on Computer Music (16. : 2017 : São Paulo)  
Proceedings [recurso eletrônico] do 16º Brazilian Symposium on Computer Music (SBCM), 03 a  
06 de setembro de 2017, São Paulo, SP / editado por Luiz Naveda ... [et al.]. – São Paulo: USP, 2017.

Disponível em: <http://compmus.ime.usp.br/sbcm/2017/assets/SBCM2017Proceedings.pdf>  
ISSN: 2175-6759  
ISBN: 978-85-76694-76-2

1. Música por computador. 2. Arte e tecnologia. 3. Multimídia (Arte). I. Naveda, Luiz (Ed.). II.  
Título.

CDU: 78:004



## Proceedings of the 16th Brazilian Symposium on Computer Music

ISSN 2175-6759

<http://compmus.ime.usp.br/sbcm/>

Edited by:

Luiz Naveda, State University of Minas Gerais  
Rogério Costa, University of São Paulo  
Marcelo Queiroz, University of São Paulo  
Flávio Schiavoni, Federal University of São João del Rey  
Rodrigo Schramm, Federal University of Rio Grande do Sul  
Tiago Tavares, State University of Campinas

Permission to make digital or hard copies of all part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© Brazilian Computer Society (Sociedade Brasileira de Computação – SBC)

CNPJ nº 29.532.264/0001-78

<http://www.sbc.org.br>

Av. Bento Gonçalves, 9500 – Setor 4 – Sala 116 – Prédio 43424 – Agronomia  
CEP 91501-970 – Porto Alegre – RS, Brasil

SBCM 2017 is organized by

CE-CM: Computer Music Interest Group of the Brazilian Computing Society  
University of São Paulo  
Institute of Mathematics and Statistics (IME)  
School of Communication and Arts (ECA)  
Sonology Research Center (NuSom)

and has the support and promotion of the following institutions

São Paulo Research Foundation (FAPESP)  
Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq)  
Brazilian Computing Society (SBC)  
State University of Minas Gerais (UEMG)  
Federal University of São João del Rey (UFSJ)  
Federal University of Rio Grande do Sul (UFRGS)  
State University of Campinas (UNICAMP)





# Introduction

Welcome to the 16th edition of SBCM! The Brazilian Symposia on Computer Music are thriving and exciting venues for sharing ideas about recent developments in the fields of computer music, sound and music processing, music information retrieval, computational musicology, multimedia performance and many other things related to art, science and technology. The event takes place at the University of São Paulo, Brazil, from September 3rd to September 6th 2017, and has as Keynote Speakers Xavier Serra (Director of the Music Technology Group at the Universitat Pompeu Fabra in Barcelona), Emiliós Cambouropoulos (Member of the Cognitive and Computational Musicology Group at the Aristotle University of Thessaloniki) and Damián Keller (Founder of the Amazon Center for Music Research at the Federal University of Acre). The program of the conference includes keynote talks, oral presentations of music and technical papers, poster discussion sessions, discussion panels and concerts, providing plenty of opportunities for interaction and discussion as a way to foster collaborations and novel ideas for the critical problems of our related fields.

The present volume features the contributions presented at SBCM 2017, including full technical papers, full music papers, posters and art, which express the ongoing exchange taking place among the fields of music, computer science and engineering, among others, and their contributions to the advancement of scientific and artistic practices. Since 1994, the Brazilian Computer Music Symposia have provided a window into the state-of-the-art developments in this rich intersection of ideas, interests and competences which converge into interdisciplinary work. The 2017 edition emphasizes current research in audio open-software resources, ubiquitous music, music information retrieval and music cognition, featuring contributions from Latin America and other research networks around the world. In this edition, the call for art, music and scientific contributions received 50 submissions of full papers, 12 poster submissions, and 24 art submissions. The double-blind, peer-reviewed process involved 71 reviewers, contributing with almost 260 evaluations that lead to an acceptance rate of 45% for full technical papers and 57% overall. Such an outstanding response of the participant interdisciplinary research communities is represented in the technical and artistic program, being complemented with the full versions of the selected submissions in these proceedings. We hope you will benefit from it!

The SBCM 2017 Organizing Committee

Marcelo Queiroz, University of São Paulo (General Chair)

Luiz Naveda, State University of Minas Gerais (Technical Papers Chair)

Rogério Costa, University of São Paulo (Music Chair)

Flávio Schiavoni, Federal University of São João del Rey

Rodrigo Schramm, Federal University of Rio Grande do Sul

Tiago Tavares, State University of Campinas





# Committees

## Organizing Committee

Marcelo Queiroz, University of São Paulo (General Chair)  
Luiz Naveda, State University of Minas Gerais (Technical Papers Chair)  
Rogério Costa, University of São Paulo (Music Chair)  
Flávio Schiavoni, Federal University of São João del Rey  
Rodrigo Schramm, Federal University of Rio Grande do Sul  
Tiago Tavares, University of Campinas

## Local Arrangements Committee

Roberto Piassi Passos Bodo, University of São Paulo (coordinator & webmaster)  
Shayenne Moura, University of São Paulo  
Felipe Felix, University of São Paulo  
Felipe Moreira, University of São Paulo  
Pedro Marcondes, University of São Paulo

## Scientific Committee

Adolfo Maia Jr., University of Campinas  
Alexandre Porres, University of São Paulo  
Aluizio Oliveira, Federal University of Minas Gerais  
Bernardo Barros, New York University  
Carolina Brum Medeiros, FlipRL, University of São Paulo, Google  
Clayton Mamedes, University of Campinas  
Davi Mota, Federal University of Minas Gerais  
Emilios Cambourpoulous, Aristotle University of Thessaloniki  
Evandro Miletto, Federal Institute of Rio Grande of Sul  
Fabio Kon, University of São Paulo  
Flávio Schiavoni, Federal University of São João Del Rei  
Flávio Soares Corrêa da Silva, University of São Paulo  
Isabel Barbancho, University of Malaga  
Jean Bresson, IRCAM  
Jean-Pierre Briot, LIP6 (CNRS - Univ. Paris 6) & PUC-Rio  
Joe Timoney, Maynooth University  
Jose Fornari, University of Campinas  
José Padovani, University of Campinas  
João Monnazzi, University of São Paulo  
Juliano Foleiss, Federal Technological University of Paraná  
Leandro Costalonga, Federal University of Espírito Santo  
Luc Nijs, Ghent University  
Luciano Flores, QI Escolas e Faculdades  
Luis Jure, Universidad of la República  
Luiz Biscainho, Federal University of Rio of Janeiro  
Luiz Naveda, State University of Minas Gerais  
Manuel Falleros, University of Campinas  
Marcelo Pimenta, Federal University of Rio Grande of Sul  
Marcelo Queiroz, University of São Paulo  
Marcelo Wanderley, McGill University  
Marcos Laia, Federal University of São João Del Rei  
Martín Rocamora, Universidad of la República  
Maurício Loureiro, Federal University of Minas Gerais  
Micael Antunes da Silva, University of São Paulo  
Regis Faria, University of São Paulo  
Rodrigo Schramm, Federal University of Rio Grande of Sul  
Sever Tipei, University of Illinois School of Music  
Stella Pashalidou, Technological Educational Institute of Crete  
Sérgio Freire, Federal University of Minas Gerais  
Tairone Magalhaes, Federal University of Minas Gerais  
Tiago Tavares, University of Campinas  
Victor Lazzarini, Maynooth University

## Music Committee

Alessandra Bochio, University of São Paulo  
Bruno Ruviaro, Santa Clara University  
Damian Keller, Federal University of Acre  
David Borgo, University of California San Diego  
Duncan Williams, Plymouth University  
Eloi Fritsch, Federal University of Rio Grande do Sul  
Federico Visi, Universität Hamburg  
Felipe Castellani, University of Campinas  
Fernando Iazzetta, University of São Paulo  
Flávio Schiavoni, Federal University of São João Del Rei  
Franziska Schroeder, Queen's University Belfast  
Isabel Nogueira, Federal University of Rio Grande do Sul  
Ivan Eiji Yamauchi Simurra, University of São Paulo  
James Correa, Federal University of Pelotas  
Jônatas Manzolli, University of Campinas  
José Padovani, University of Campinas  
Julian Arango, Caldas University  
Luciano Zanatta, Federal University of Rio Grande do Sul  
Luiz Casteloos, Federal University of Juiz de Fora  
Luke Dahl, University of Virginia  
Marcelo Gimenes, Plymouth University  
Marcus Bittencourt, State University of Maringá  
Ricardo Bordini, Federal University of Bahia  
Ricardo Dal Farra, Concordia University  
Rodolfo Coelho de Souza, University of São Paulo, Ribeirão Preto  
Rodrigo Schramm, Federal University of Rio Grande do Sul  
Rogerio Constante, Federal University of Pelotas  
Rogerio Costa, University of São Paulo  
Stéphan Schaub, University of Campinas



# Table of Contents

## Technical Papers

A Domain Specific Language For Drum Beat Programming	
André Rauber Du Bois, Rodrigo Ribeiro	1
A Probabilistic Model For Recommending Music Based on Acoustic Features and Social Data	
Rodrigo C. Borges, Marcelo Queiroz	7
A Score-Informed Approach for Pitch Visualisation of a Cappella Vocal Quartet Performances	
Rodrigo Schramm, Helena de Souza Nunes, Emmanouil Benetos	13
aAaA: an attribute aware abstraction architecture allowing arbitrary argument assignment in Pure Data	
José Henrique Padovani	19
Challenges for a Second Decade of Ubimus Research: Metaphors for Creative Action (part 1)	
Damián Keller	25
Design and implementation of an open-source subtractive synthesizer on the Arduino Due platform	
Rodolfo Pedó Pirotti, Marcelo Johann, Marcelo Pimenta	33
ELSE Library for Pure Data	
Alexandre Torres Porres	41
Gestures of Body Joints, Musical Pulses and Laban Effort Actions: Towards an Interactive Tool for Music and Dance	
Leandro Souza, Sérgio Freire	49
Impact of Algorithmic Composition on Player Immersion in Computer Games: A Case Study Using Markov Chains	
Raul Paiva de Oliveira, Tiago Fernandes Tavares	55
Impact of Genre in the Prediction of Perceived Emotions in Music	
Felipe Souza Tanios, Tiago Fernandes Tavares	63
Sounderfeit: Cloning a Physical Model with Conditional Adversarial Autoencoders	
Stephen Sinclair	67
Synesthesia Add-on: a Tool for HTML Sonification	
Roberto Piassi Passos Bodo, Flávio Luiz Schiavoni	75
Technology Enhanced Learning of Expressive Music Performance	
Rafael Ramirez, Fabio Ortega, Sergio Giraldo	81
The Million Playlists Songs Dataset: a descriptive study over multiple sources of user-curated playlists	
Felipe Falcão, Daniel Mélo	86
Timbre spaces with sparse autoencoders	
Pablo E. Riera, Manuel C. Eguía, Matías Zabaljáuregui	93
Vivace: a collaborative live coding language and platform	
Vilson Vieira, Guilherme Lunhani, Geraldo Magela de Castro Rocha Junior, Caleb Mascarenhas Luporini, Daniel Penalva, Ricardo Fabbri, Renato Fabbri	99
Web Audio application development with Mosaiccode	
Flávio Luiz Schiavoni, Luan Luiz Gonçalves, André Lucas Nascimento Gomes	107

## Music Papers

AirQ Sonification as a context for mutual contribution between Science and Music	
Julian Arango	115
An Analysis of <i>Desdobramentos do continuo</i> for violoncello and live electronics performed by audio descriptors	
Danilo Rosseti, Willian Teixeira, Jónatas Manzolli	123
Communicating a World View: figer, a Manifold Composition	
Sever Tipei	131
The Maxwell Demon: Comprovisation in Ecologically Grounded Creative Practice	
Luzilei Aliel, Damián Keller, Rogério Costa	137
Web Orchestra Studio: a real-time interactive platform for music and education	
Juliano Kestenber, Vitor Rolla, Djalma Lúcio, Luiz Velho	144

## Posters

A tool for the musical education of Deaf people	
Erivan Duarte, Tiago Tavares	152

An Algorithm for Guiding Expectation in Free Improvisational Solo Performances	
José Fornari . . . . .	154
An open dataset for vocal music transcription	
Marcos Weitowitz, Helena Souza Nunes, Rodrigo Schramm . . . . .	156
Complex networks of chord transitions in Alexander Scriabin's piano pieces	
Augusto Paladino, Bruno Mesz, Juan Pégola . . . . .	158
Developing a set of applications for music creation using low-cost brain-computer interfaces	
Guilherme Feulo do Espírito Santo, Marcelo Queiroz . . . . .	160
Live Coding Console with Remote Audience into the web	
Guilherme Lunhani, Flávio Schiavoni . . . . .	162
Melody and accompaniment separation using enhanced binary masks	
Shayenne Moura, Marcelo Queiroz . . . . .	164
Methods on Composer Identification Using Markov Chains	
Adilson Neto, Rodrigo Pereira . . . . .	166
New developments on the augmentation of a classical guitar: Addition of embedded sound synthesis and OSC communication over network	
Eduardo Meneses, Marcelo Wanderley . . . . .	168
Pedal Board Approach to Sound Effects Customization	
Thiago Felipe de Miranda Arcanjo, Franklin Magalhães Ribeiro Junior, Tarcisio da Rocha . . . . .	170
T2M/M2T: Sonifying Text and Textualizing Sound Using Audio Tags	
Bruno Mesz, Lucas Samaruga . . . . .	172

## Art

Allure, a machinic performance	
André Martins . . . . .	176
Colour Etude I	
Omar Peracha . . . . .	178
Desdobramentos do contínuo for violoncello and live electronics	
Danilo Rosseti, Willian Teixeira . . . . .	180
figer, Computer-assisted Composition for Computer-generated Sounds (fixed media)	
Sever Típei . . . . .	182
“Era como se estivéssemos vivos”, for alto saxophone and live electronic sounds	
Rodolfo Valente . . . . .	184
Lignes et Pointes - étude pour la décomposition en deux parties d'une oeuvre de Joan Miró	
Antonio D'Amato . . . . .	185
Puzzle Pieces	
Paul Schuette . . . . .	186
Rare yet soft	
Kyong Mee Choi . . . . .	207
Suíte [en]Quadrada	
Yonara Dantas, Miguel Antar, Luzilei Aliel . . . . .	208
The Maxwell Demon	
Luzilei Aliel . . . . .	210

## Summary

Author Index . . . . .	212
Institution Index . . . . .	213

# Technical Papers



# A Domain Specific Language For Drum Beat Programming

André Rauber Du Bois<sup>1</sup>, Rodrigo Ribeiro<sup>12</sup>

<sup>1</sup>PPGC - Universidade Federal de Pelotas, Pelotas - RS

<sup>2</sup>PPGCC - Universidade Federal de Ouro Preto, Ouro Preto - MG

dubois@inf.ufpel.edu.br, rodrigo@decsi.ufop.br

## Abstract

This paper describes HDrum, a Domain Specific Language for writing drum patterns. Programs written in HDrum look similar to the grids, available in sequencers and drum machines, used to program drum beats, but as the language has an inductive definition we can write abstractions to manipulate drum patterns. HDrum is embedded in the Haskell functional programming language, hence it is possible to implement Haskell functions that manipulate patterns generating new patterns. The paper also presents a case study using HDrum, an implementation of *The Clapping Music*, a minimalistic music written by Steve Reich in 1972. The HDrum language is currently compiled into midi files.

## 1. Introduction

This paper presents the HDrum language, a domain specific language for drum pattern programming embedded in the Haskell functional programming language. Simple drum pattern programming in HDrum looks like an ASCII version of the grids for drum beat programming available in sequencers and drum machines. But the constructors used to describe patterns have a well defined inductive semantics which leads to interesting properties and allows the definition of functions over patterns and multi-tracks. HDrum provides two abstractions, patterns and tracks, the first is used to describe a drum beat and the second to associate an instrument to a pattern. HDrum is an algebra that defines the sets of patterns and tracks and also the functions that operate on these sets. Mainly HDrum provides operators for repetition, sequencing and parallel composition of patterns and tracks.

As the DSL is embedded in Haskell, it is possible to use all the power of functional pro-

gramming in our benefit to define new abstractions over drum patterns. To understand the paper the reader needs no previous knowledge of Haskell, although some knowledge of functional programming and recursive definitions would help. We try to introduce the concepts and syntax of Haskell needed to understand the paper as we go along.

Although HDrum and its prototype implementation are only designed for drum beat programming, the abstractions provided by the language can be applied for general music programming as well.

The paper is organized as follows. First we describe the main constructors for patterns (Section 2.1) and tracks (Section 2.2) design and their basic operations. Next, we examine the important abstraction of track composition (Section 2.3). Section 2.4, provides a discussion on simple algebraic properties of the language, e.g., the notion of equality, associativity, etc. The compilation of HDrum into midi files is explained in Section 3. A case study, the implementation of the minimalistic music *The Clapping Music*, is presented in Section 4. Finally, related work, conclusions and future work are discussed.

## 2. HDrum: Drum patterns for Haskell

### 2.1. Simple Drum Patterns

HDrum is an algebra (i.e., a set and the respective functions on this set) for drum pattern programming. The set of all drum patterns can be described inductively as an algebraic data type in Haskell:

```
data DrumPattern = X | O |
  DrumPattern :| DrumPattern
```

The word `data` creates a new data type, in this case, `DrumPattern`. This definition says

that a drum pattern can be either a hit (X), a rest (O), or a sequential composition of patterns using the operator (`:|`), that takes as arguments two drum patterns and returns a new drum pattern.

As an example, we can define two 4/4 drum patterns, one with a hit in the 1st beat called `kick` and another that hits in the 3rd called `snare`.

```
kick :: DrumPattern
kick = X :| O :| O :| O
```

```
snare :: DrumPattern
snare = O :| O :| X :| O
```

The symbol (`::`) is used for type definition in Haskell, and can be read as *has type*, e.g. `kick` has type `DrumPattern`.

As `DrumPattern` is a recursive data type, it is possible to write recursive Haskell functions that operate on drum patterns. For example, usually a certain pattern is repeated many times in a song, and a repeat operator (`.*`) for patterns can be defined as follows:

```
(.*) :: Int -> DrumPattern
      -> DrumPattern
```

```
1 .* p = p
n .* p = p :| (n-1) .* p
```

The repeat operator takes as arguments an integer `n` and a drum pattern `p`, and returns a drum pattern that is a composition of `n` times the pattern `p`. As can be seen in the previous example, the composition operator can combine drum patterns of any size and shape, e.g.:

```
hihatVerse :: DrumPattern
hihatVerse = 8 .* (X :| O :| X :| O)
```

```
hihatChorus :: DrumPattern
hihatChorus = 4 .* (X :| X :| X :| X)
```

```
hihatSong :: DrumPattern
hihatSong = hihatVerse :|
            hihatChorus :|
            hihatVerse :|
            hihatChorus
```

or simply:

```
hihatSong :: DrumPattern
hihatSong = 2 .* (hihatVerse :|
                 hihatChorus)
```

In order to make any sound, a drum pattern must be associated to an instrument hence generating a `Track`, as explained in the next section.

## 2.2. Tracks

A track is the `HDrum` abstraction that associates an instrument to a pattern. The `Track` data type is also defined as an algebraic type in Haskell:

```
data Track =
  MakeTrack Instrument DrumPattern
  | Track :|| Track
```

A simple track can be created with the `MakeTrack` constructor, which associates an `Instrument` to a `DrumPattern`. A `Track` can also be the *parallel* composition of two tracks, which can be obtained with the `:||` operator. In the current implementation of the language, the instruments available are the different drum sounds of the midi protocol [1]. `Instruments` is also defined as an algebraic data type listing all possible instruments:

```
data Instrument = AcousticBassDrum
  | BassDrum | SideStick
  | AcousticSnare | HandClap
  | (...)
```

Now, we can use the previously defined patterns `kick` and `snare` to create tracks:

```
kickTrack :: Track
kickTrack = MakeTrack BassDrum kick
```

```
snareTrack :: Track
snareTrack =
  MakeTrack AcousticSnare snare
```

and also multi-tracks:

```
rockMTrack :: Track
rockMTrack =
  kickTrack :||
  snareTrack :||
  MakeTrack ClosedHiHat (X:|X:|X:|X)
```

## 2.3. Composing Tracks

The `:||` operator allows the parallel composition of `Tracks`, i.e., adding an extra track to a multi-track song. But what if we want to compose tracks in sequence, e.g., we have different multi-tracks for the introduction, verse and chorus, and want to combine them in sequence to form a complete song?

One problem that we need to deal with are the different sizes of patterns in a multi-track. The size of a multi-track, is the size of its largest pattern. It is important to notice that when composing tracks, we assume that smaller patterns have rest beats at their

```

track1 =
  MakeTrack BassDrum          (X)
  :|| MakeTrack AcousticSnare (O :| O :| X)
  :|| MakeTrack ClosedHiHat   (X :| X :| X :| X)

track2 =  MakeTrack BassDrum  (X :| O :| O :| O)
  :|| MakeTrack AcousticSnare (O :| O :| X :| O)
  :|| MakeTrack ClosedHiHat   (X :| O :| X )
  :|| MakeTrack Cowbell       (X)

track1track2 =
  MakeTrack BassDrum          (X :| O :| O :| O :| X :| O :| O :| O)
  :|| MakeTrack AcousticSnare (O :| O :| X :| O :| O :| O :| X :| O)
  :|| MakeTrack ClosedHiHat   (X :| X :| X :| X :| X :| O :| X )
  :|| MakeTrack Cowbell       (O :| O :| O :| O :| X )

track2track1 =
  MakeTrack BassDrum          (X :| O :| O :| O :| X)
  :|| MakeTrack AcousticSnare (O :| O :| X :| O :| O :| O :| X :| O)
  :|| MakeTrack ClosedHiHat   (X :| O :| X :| O :| X :| X :| X :| X)
  :|| MakeTrack Cowbell       (X)

track1twice =
  MakeTrack BassDrum          (X :| O :| O :| O :| X)
  :|| MakeTrack AcousticSnare (O :| O :| X :| O :| O :| O :| X )
  :|| MakeTrack ClosedHiHat   (X :| X :| X :| X :| X :| X :| X :| X)

track2twice =
  MakeTrack BassDrum          (X :| O :| O :| O :| X :| O :| O :| O)
  :|| MakeTrack AcousticSnare (O :| O :| X :| O :| O :| O :| X :| O)
  :|| MakeTrack ClosedHiHat   (X :| O :| X :| O :| X :| O :| X)
  :|| MakeTrack Cowbell       (X :| O :| O :| O :| X)

```

**Figure 1: Composing tracks**

end, hence all patterns are assumed to have the size of the largest pattern in a multi-track. We can define this concepts formally with the following recursive functions:

```
lengthDP :: DrumPattern -> Int
lengthDP O = 1
lengthDP X = 1
lengthDP (X:|p) = 1 + lengthDP p
lengthDP (O:|p) = 1 + lengthDP p
lengthDP (x:|y) = lengthDP x +
                  lengthDP y

lengthTrack :: Track -> Int
lengthTrack (MakeTrack _ dp) =
  lengthDP dp
lengthTrack ((MakeTrack _ dp):|t) =
  max (lengthDP dp) (lengthTrack t)
```

Where `lengthDP` recursively calculates the size of a drum pattern, and `lengthTrack` finds out the size of the largest pattern in a track, i.e., the size of the track.

HDrum provides two constructs for composing tracks in sequence, a repetition operator `|*` and a sequencing operator `|+`. The repetition operator is similar to `.*` but operates on all patterns of a multi-track:

```
|* :: Int -> Track -> Track
```

It that takes an integer `n` and a multi-track `t` and repeats all patterns in all tracks `n` times adding the needed rest beats at the end of smaller tracks.

The semantics of composing two multi-tracks `t1` and `t2`, i.e., `t1 |+` `t2` is as follows:

- First we add rest beats to the end of each track in `t1` that has matching instruments with tracks in `t2`, so that all those tracks have the same size as the largest pattern in `t1`
- Then, for all patterns `p1` in `t1` and `p2` in `t2` that have the same instrument `i`, we generate a new track `MakeTrack i (p1:|p2)`
- Finally, we add a pattern of rests the size of `t1`, to the beginning of all tracks in `t2` that were not composed with tracks in `t1` in the previous step

Hence the size of the composition of two tracks `t1` and `t2` is sum of the size of the largest pattern in `t1` with the largest pattern in `t2`.

In Figure 1 it is possible to see two tracks with different sizes of patterns inside (`track1` and `track2`) and their compositions, `track1track2` is the same as `track1 |+` `track2` and `track2track1` is the same as `track2 |+` `track1`. The `track1twice track` is equivalent to `2 |*` `track1` and `track2twice` is equivalent to `2 |*` `track2`.

## 2.4. Algebraic Properties

In this section we discuss some algebraic properties of drum patterns and tracks. Data types `DrumPattern` and `Track` provide a syntactic representation of music data, which allows different sound renderings. As an example, let `v1`, `v2` and `v3` be any value of type `DrumPattern`. Then, one should expect that `v1 :| (v2 :| v3)` will have the same meaning as `(v1 :| v2) :| v3`, i.e. sequential composition is an associative operation. From a semanticist point of view, such values are different, since they represent distinct syntactic entities, but they can have the same meaning using an appropriate notion of equality.

We consider two `DrumPatterns` or `Tracks` equal if they produce the same music. In order to define such equality precisely, we will need an algebra of drum music. Formally, an *algebraic structure*  $\langle S, op_1, op_2, \dots, op_{n-1} \rangle$  is a  $n$ -uple formed by a non-empty carrier set  $S$  and operations over it. The algebraic structure of drum sounds is formed by a set  $\mathcal{D}$  of drum music values with a distinct value  $\epsilon$  to denote rest, functions for sequential and parallel composition,  $\oplus$  and  $\parallel$  respectively, a function for translating hits of a given instrument to its correspondent music value, namely  $\llbracket \_ \rrbracket_I : \text{Instrument} \rightarrow \mathcal{D}$  and an equivalence relation between  $\mathcal{D}$  elements denoted as  $v \equiv v'$ , for some  $v, v' \in \mathcal{D}$ . We assume that  $\parallel$  is commutative and both  $\oplus$  and  $\parallel$  are associative operators, with identity  $\epsilon^0$  for  $\oplus$ ,  $\epsilon^n$  for  $\parallel$ , where  $t^n$  denotes the parallel composition of  $n$  copies of  $t$ . When  $n = 0$ ,  $t^0$  denotes a rest of duration 0. We let  $\mathcal{P}$  denote the set of pairs formed by a value of type `Instrument` and a value of type `DrumPattern`.

Translation of patterns and tracks is easily defined by recursion as follows as in Figure 2.

Using the semantics, we can define an equivalence relation for HDrum values. Let  $v_1$  and  $v_2$  be two drum patterns or tracks. We say that they are equivalent,  $v_1 \approx v_2$ , if and only if  $\llbracket v_1 \rrbracket \equiv \llbracket v_2 \rrbracket$ , where  $\llbracket v_1 \rrbracket$  denotes the semantics for patterns or tracks, respectively. Using this equivalence relation, we can check



$$\begin{aligned}
\llbracket -, - \rrbracket_D & : \mathcal{P} \rightarrow \mathcal{D} \\
\llbracket i, X \rrbracket_D & = \llbracket i \rrbracket_I \\
\llbracket i, O \rrbracket_D & = \epsilon \\
\llbracket i, d_1 : | d_2 \rrbracket_D & = \llbracket i, d_1 \rrbracket_D \oplus \llbracket i, d_2 \rrbracket_D \\
\llbracket - \rrbracket_T : \text{Track} & \rightarrow \mathcal{D} \\
\llbracket \text{MakeTrack } i \text{ d} \rrbracket_T & = \llbracket i, d \rrbracket_D \\
\llbracket t_1 : || t_2 \rrbracket_T & = \llbracket t_1 \rrbracket_T || \llbracket t_2 \rrbracket_T
\end{aligned}$$

**Figure 2: Semantics of HDrum.**

that HDrum patterns enjoys some algebraic properties. We list some of them below:

- **Associativity:** sequential and parallel composition are associative: For all  $p_1, p_2$  and  $p_3$  of type `DrumPattern`, we have  $p_1 : | (p_2 : | p_3) \approx (p_1 : | p_2) : | p_3$ . For all  $t_1, t_2$  and  $t_3$  of type `Track`, we have  $t_1 : | (t_2 : | t_3) \approx (t_1 : | t_2) : | t_3$ .
- **Identity:**  $O^0$  is the identity for sequential composition.
- **Commutativity of parallel composition:** for all tracks  $t_1$  and  $t_2$ , we have that  $t_1 : || t_2 \approx t_2 : || t_1$ .

Such properties are easily proved by induction over the structure of `DrumPatterns` and `Tracks`, using the respective properties of  $\oplus$  and  $||$  operators.

### 3. Compiling HDrum into midi files

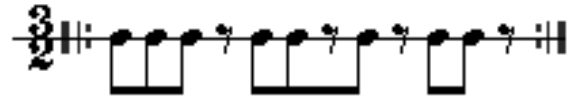
Midi is a standardized protocol to transmit real time information for the playback of a piece of music. The protocol defines a collection of messages that can be used to play music and for communication between Midi devices. Midi files contain the description of a piece of music, i.e., information such as what notes are played, when they are played, for how long and how loud. Specifically for the implementation of HDrum the important messages are the `NOTE ON` and `NOTE OFF` which tell when to start and stop playing a sound. Basically, the HDrum compiler traverses the data structure of patterns and tracks generating the appropriate `NOTE ON` and `NOTE OFF` messages for the drum instruments specified at tracks. We used Haskell's `Codec.Midi` library to generate the files. This library provides an algebraic data type

defining all midi messages and handles the generation of binary files from a list of midi messages.

### 4. Case study: The clapping music

Minimalistic music is a type of art music, created in the early sixties, that uses minimal musical material. A famous piece created in this style is *The Clapping Music*, written by Steve Reich in 1972. The song was written to be performed by two people clapping the pattern in Figure 3. After 8 bars, one of the players shifts the pattern one eighth note to the right. The player keeps shifting every eight bars until his pattern meets again the pattern of the first player. A video of Steve Reich himself playing the piece can be seen in [2].

The interesting thing about this song is that shifting bits is a very common operation in computer science, so we can actually program the song in HDrum.

**Figure 3: The clapping music**

The initial pattern of the song, can be implemented in HDrum as follows:

```

clappingPat :: DrumPattern
clappingPat = X :| X :| X :| O :|
              X :| X :| O :| X :|
              O :| X :| X :| O

```

We can shift a pattern using the following function:

```

shiftPat :: DrumPattern -> DrumPattern
shiftPat O = O
shiftPat X = X
shiftPat (O:|p) = p :| O
shiftPat (X:|p) = p :| X
shiftPat (p1:|p2) = (tailDP p1) :|
                    p2:|(headDP p1)

```

If the pattern to be shifted is just a simple hit or rest, `shiftPat` does nothing. If the first element of the pattern is a hit or rest, then we simply move it to the end. If the composed pattern is formed by two more complex patterns ( $p_1$  and  $p_2$ ), we remove the first beat of  $p_1$  and add it to the end of the pattern using `headDP` to get the first element of the pattern.

Now, it is possible to write a function that creates a pattern by shifting the beats of an initial pattern:

```
shiftMany :: Int -> Int
           -> DrumPattern
           -> DrumPattern
shiftMany 1 t p = t .* (shiftPat p)
shiftMany n t p =
  t .* shifted
  :| shiftMany (n-1) t shifted
  where shifted = shiftPat p
```

The `shiftMany` function takes two integers (`n` and `t`) and a pattern `p` as arguments and returns a new pattern that shifts `p` `n` times, and each shifted version of `p` is repeated `t` times.

Now it is possible to implement the song. The first pattern, the one that is not shifted, is just a repetition of `clappingPat`. The second pattern, starts with the basic pattern and then is shifted every eight bars. It must shift 12 times in order to become the initial pattern again. Hence we have:

```
fstPatCS :: DrumPattern
fstPatCS = 104 .* clappingPat

sndPatCS :: DrumPattern
sndPatCS = 8 .* clappingPat
  :| shiftMany 12 8 clappingPat

and the final clapping song becomes:

clappingSong :: Track
clappingSong =
  MakeTrack HandClap fstPatCS
  :|| MakeTrack HandClap sndPatCS
```

## 5. Related Works

There has been a lot of work on designing programming languages for computer music. Due to lack of space, we discuss here the ones that are closer to HDrum. There are many DSLs for computer music based on functional languages, e.g. [3, 4, 5, 6, 7]. These languages usually provide means for playing notes and composing the sounds generated in sequence and in parallel. In these languages the programmer can write a sequence of notes and rests, and these sequences can also be combined in parallel. In HDrum, instead of having different notes in the same track, each track indicates when a single note is played, i.e., It is the repetition pattern of a single note, similar to what happens in grids of a drum machine. Although the symbols used in HDrum have semantic meaning, visually programs look like

an ASCII version of the grids for writing drum beats available in modern sequencers. We believe that this approach makes it easier for someone that is used with sequencer tools to write simple tracks in HDrum with little knowledge of functional programming. Furthermore, as patterns are not associated with notes, patterns can be reused with different instruments when needed.

## 6. Conclusions and Future Work

This paper has described the HDrum language for drum beat programming and its abstractions. Although HDrum was designed for percussion instruments, the ideas presented here can be easily adapted for general music programming. Our ultimate goal is to design a full language for live music programming. The current implementation of HDrum can be found in [8].

## Acknowledgement

This work was supported by CAPES/Brasil (Programa Nacional de Cooperação Acadêmica da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior).

## References

- [1] Midi Instruments. <https://www.midi.org/specifications/item/gm-level-1-sound-set>, May 2017.
- [2] The Clapping Music. <https://www.youtube.com/watch?v=hH1j06bMH-DQ&t=115s>, May 2017.
- [3] Alex McLean. Making programming languages to dance to: Live coding with tidal. In *FARM 2014*. ACM, 2014.
- [4] Paul Hudak, Tom Makucevich, Syam Gadde, and Bo Whong. Haskore music notation: An algebra of music. *J. of Functional Programming*, 6(3), May 1996.
- [5] H. Thielemann. Audio processing using haskell. In *DAFx'04*, 2004.
- [6] Paul Hudak and David Janin. Tiled polymorphic temporal media. In *FARM 2014*. ACM, 2014.
- [7] Paul Hudak. An algebraic theory of polymorphic temporal media. In *PADL*, 2004.
- [8] HDrum Language. <https://sites.google.com/site/hdrumlanguage/>, May 2017.

# A Probabilistic Model For Recommending Music Based on Acoustic Features and Social Data

Rodrigo C. Borges<sup>1</sup>, Marcelo Queiroz<sup>1\*</sup>

<sup>1</sup>Grupo de Computação Musical - IME - USP  
Av. Prof. Luciano Gualberto, 158, tv. 3 – 05508-900 São Paulo, SP

rcborges@ime.usp.br, mqz@ime.usp.br

## Abstract

The “Cold Start” problem is a well known issue in Collaborative Filtering recommendation systems, associated to the moment when a new item or user is added to a given collection, because the system has no historical information of interaction between existing and new elements and it still need to incorporate these elements into the recommendation algorithm. This work addresses one possible solution for the case where new songs are added to a dataset of a music recommendation system, by proposing a probabilistic model for inference based on the songs’ acoustic/timbre features. This model was first proposed for tagging music with semantic labels but is here suggested as being suitable for predicting user interactions with new songs. The experiments were conducted using a selection of Brazilian popular music and the results show that the proposed method compares favorably to Logistic Regression.

## 1. Introduction

Recommending music automatically has become a popular issue in the last decades since a huge amount of digital media became available as on-line services [1, 2]. The most common technique used today for this purpose is called Collaborative Filtering [3], and it works by matching similar user listening profiles. If user A listened to a song that user B with a similar listening profile hasn’t, it is assumed that there is a high probability that user B would react positively to this song. But this approach has a weakness known as the “cold start” problem, which corresponds to a new song with no records of

having being listened by existing users or a new user with no records of having listened to existing songs.

One of the possible solutions for this problem is to combine acoustic features extracted from the songs and past listening behavior, hopefully finding a statistical representation of the timbre content of the listened songs [4] that would somehow correlate with listening habits. If a new song is added to the set and its acoustic content is close enough to what has been learned by the recommender as part of a user’s listening habit, then it might be considered as a recommendation candidate.

In this paper we apply a probabilistic model named Codeword Bernoulli Average Model [5] for predicting listening behaviors to new songs. This model was first proposed for tagging music with semantic labels, and attempts to predict the probability that a binary tag applies to a song, based on a vector-quantized representation of that song’s audio (Figure 1). This is achieved through automatically learning a latent variable that represents some statistical relationship between the audio and the tag, which in this case are timbre representations and user listening behaviors.

This text is structured as follows. We start briefly presenting previous probabilistic models proposed for the same problem. The dataset used in our experiment is then described, namely the songs presented to the listeners and from which acoustic features were extracted. Then we present the experimental methodology, detailing the procedure of collecting listening data, the application used and how this information was stored. Feature extraction is explained in detail as well as the vector-quantized representation used for the model. The Codeword Bernoulli

\*The second author acknowledges funding received from CNPq.

		Centroids					Users				
Songs (Train)		159	415	...	998	13	1	0	...	0	0
		292	174	...	229	28	0	1	...	0	0
		334	13	...	19	543	1	1	...	1	0
		122	0	...	711	43	0	0	...	0	1
		1	489	...	5	543	0	1	...	0	0
		488	43	...	449	54	1	0	...	0	1
Songs (Test)		150	315	...	990	22	?	?	...	?	?
		292	174	...	229	547	?	?	...	?	?
		334	134	...	119	5	?	?	...	?	?

**Figure 1: An Illustration of the problem of retrieving listening data from acoustic features.**

Average Model is then briefly described, and we discuss how to apply it to sparse listening data. The evaluation of the model is presented next, along with a comparison to a Logistic Regression baseline, discussing how far this approach can be further explored.

## 2. Related Work

Some efforts on using acoustic features combined with collaborative filtering were already reported, some of them applying probabilistic modeling. Yoshii et al. [6] has proposed using GMM for representing the MFCC information, and also an e-commerce interaction database as corresponding to social data. A group of latent variables were proposed as corresponding to genres, among which the user would choose, and from which a piece of music was stochastically selected. Pseudo-genres are considered as providing recommendation diversity but also as differing from the kind of prediction desired here.

Campos et al. [4] propose a topological model based on Bayesian networks from where the degree of each recommending technique was automatically selected. This model could operate exclusively as collaborative-filtering or as content-based, using it for finding good items as well as for predicting user ratings.

Codeword Bernoulli Average Model was first presented in 2009 as a technique for automatically tagging music [5]. In this context the challenge was separated in two parts: annotating music that has no associated tags, and retrieving songs from a given tag. In both cases a subset should be returned sorted by relevance of the re-

turned items, where the ordering had no relationship whatsoever with the binary tags.

## 3. Data Set

The dataset we used in the experiment is composed of 1199 Brazilian popular songs taken from a selection known as "100 best records of Brazilian music" [7], published in 2007 by the specialized music magazine Rolling Stone and representing the opinions of 60 music researchers, producers and journalists based on how influential they thought these records were to others artists. The recordings release dates vary from 1950 to 2003, which configures a heterogeneous group of music examples that should result in considerably different listening behavior patterns.

## 4. Listening Data

The listening data was collected between March and May 2017, having 10 listeners with ages varying from 25 to 60 years old. An Android application (Figure 2) was developed specifically for this experiment and when initialized, it selected randomly any song from the dataset and started playing. The user could listen to it or jump to the next song. This resulted in a sparse matrix counting how many times each user listened to each music until its end. It should be made clear that a count of 0 could either mean that the user was never exposed to a song, or that she or he skipped the song before reaching its end.

There were around 1000 complete song reproductions during this period, but only four listeners listened to a reasonable sample of the whole dataset (at least 10% of the number of songs), and for this reason these were the only subjects considered in the analysis.

The listening counting matrix was used to define a binary matrix representing which user had listened to which song to the end (regardless of how many times). This matrix relates users and songs through a binary correspondence that might be used as indicative of a user's willingness to hear to a song; again a value of 0 should

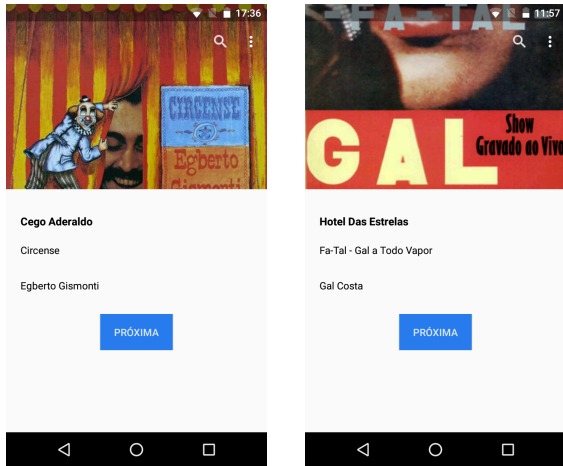


Figure 2: The application for collecting listening data.

not be understood as any negative disposition between a user and a song, because the user might not have been given the chance of listening to that song. This ambiguity is considered in the discussion and will be addressed in future work.

## 5. Feature Extraction

The MFCC acoustic descriptor was considered as a suitable representation for timbre aspects of songs, and was extracted using an open source Python library called Librosa<sup>1</sup>. The MFCC data was extracted with 13 coefficients, using windows of 2048 samples and 75% overlap between windows. As the number of windows depend on the duration of the song, and we needed a uniform representation for the dataset, we adopted a Vector Quantized Representation to solve this problem.

### 5.1. Vector Quantized (VQ) Representation

VQ is a technique first applied in data compression that considers a group of selected items for building a general feature space, building histograms to describe a large collection according to the number of members best described by each selected item. For this specific case, VQ: (i) takes all MFCC data extracted from every song in the database and define  $K$  centroids using Kmeans; (ii) takes each song and verify for each of its MFCC frames which of the  $K$  centroids is closest to it; (iii) produces a fixed size histogram for each

<sup>1</sup><https://github.com/librosa/librosa>

song, representing how many MFCC frames are best described by each centroid.

One thing that it is worth mentioning is that the feature space is defined by all MFCC vectors extracted from the dataset, and how these MFCCs are clustered in  $K$  centroids. When a new song is added to the dataset, this feature space should be recalculated, possibly defining a new group of centroids and corresponding histograms for each song.

## 6. Codeword Bernoulli Average (CBA) Model

Our CBA model assumes a collection of binary random variables  $\mathbf{y}$  with  $y_{ju} \in \{0, 1\}$ , indicating whether or not user  $u$  has listened to song  $j$ . The goal is to estimate a set of values for a Bernoulli parameters  $\beta$  that will maximize the likelihood  $p(\mathbf{y}|\mathbf{n}, \beta)$  of the listened song associated to the VQ centroids counts  $\mathbf{n}$  and the parameters  $\beta$ . It uses the Expectation Maximization (EM) algorithm for maximum likelihood estimation. Each EM iteration has two steps: first the Expectation that corresponds to:

$$h_{juk} = \begin{cases} \frac{n_{jk}\beta_{ku}}{\sum_{i=1}^K n_{ji}\beta_{iu}} & \text{if } y_{ju} = 1 \\ \frac{n_{jk}(1 - \beta_{ku})}{\sum_{i=1}^K n_{ji}(1 - \beta_{iu})} & \text{if } y_{ju} = 0, \end{cases} \quad (1)$$

followed by the Maximization step which corresponds to:

$$\beta_{ku} = \frac{\sum_j h_{juk} y_{ju}}{\sum_j h_{juk}}. \quad (2)$$

EM should stop iterating when the difference between two consecutive  $\beta$  matrices reaches a threshold value. When this happens we have found a  $\beta$  under which the training data has become more likely.

This results in a  $\beta$  matrix with dimensions given by the number of users ( $U$ ) and the number of centroids ( $K$ ), representing the statistical relationship between users and centroids through a Bernoulli distribution.

## 6.1. Generalization

The result can then be generalized to new songs by simply multiplying its VQ representation vector by the corresponding column from the  $\beta$  matrix. The probability of a new song being heard given its feature vector is given by:

$$p(y_{ju} = 1 | \mathbf{n}_j, \beta) = \frac{1}{N_j} \sum_k n_{jk} \beta_{ku} \quad (3)$$

where the normalization  $N_j$  represents how many MFCC windows were extracted from this specific song.

## 7. Evaluation

MFCC data was extracted from all 1199 songs, resulting in 10.844.508 feature vectors. These vectors were grouped in 5, 10, 25, 50, 100 and 200 centroids, in order to compare experimentally VQ representations of several orders. The listening information was then concatenated with each timbre vector in order to have all data contained in one single matrix.

Each of the following was performed independently 20 times:

- Data matrix rows were shuffled;
- Data is then split in training and test subsets corresponding to 80% and 20% of the whole set. Listening information is separated and acoustic features data is normalized by the number of MFCCs extracted of each song;
- For the Logistic Regression the first subset was used to train the model, and the second for testing. Predictions were recorded in a text file;
- For the CBA the training subset was used for learning the  $\beta$  matrix, and the test subset for generalizing through equation 3;
- Predictions were recorded in a text file;
- F-measure, Precision, Recall and AROC values were calculated comparing predictions and true test values. The threshold for considering the predicted probability as 0 or 1 was learned from the training data;

- A random vector was generated and also compared to true values;
- Performance measurements were recorded in a log file.

CBA should stop iterating the learning loop once the difference between two consecutive matrices was above 1% of the number of centroids. Logistic Regression was chosen as a baseline reference for comparison purposes only, and the results are presented in Table 1.

## 7.1. Results Discussion

Recall, precision, f-measure and area under the receiver-operator curve (AROC) are standard metrics for evaluating binary classifiers [8]. Recall is obtained as the relation between true positive and the sum of true positives and false negatives ( $R=tp/(tp+fn)$ ); Precision is the relation between true positives and sum of true positives and false positives ( $P=tp/(tp+fp)$ ); and f-measure is the harmonic mean between both ( $F=2PR/(P+R)$ ). AROC is the area under the curve representing true positive rate against the false positive rate.

The larger recall and f-measure values were obtained for the case of CBA with the lowest value for K. Here it means that 5 centroids is the best scenario where the relationship between MFCC representations and user behaviors was best captured in the  $\beta$  matrix, possibly meaning that a better prediction model would not use so many latent variables (the MFCC centroids) to express the recommendation as a function of the MFCC histograms.

We can also see that both f-measure and recall values decrease as functions of K, for both CBA and Logistic Regression. This might be interpreted in terms of a form of overfitting of the model to a given (training) dataset which is not able to perform equally well on a different (test) dataset. Overfitting would certainly explain the monotonicity of these metrics, where both f-measure and recall get progressively worse as more MFCC centroids are used to model each user's listening preference.

Precision, on the other hand does not display a very clear trend, although it is marginally higher for Logistic Regression than it is for CBA. The

		Recall	Precision	F-Measure	AROC
K=5	Log. Reg.	0.516 (0.117)	0.125 (0.016)	0.201	<b>0.525</b> (0.023)
	CBA	<b>0.677</b> (0.136)	0.119 (0.016)	<b>0.203</b>	0.513 (0.028)
K = 10	Log. Reg.	0.405 (0.070)	0.113 (0.013)	0.177	0.517 (0.022)
	CBA	<b>0.677</b> (0.119)	0.112 (0.010)	0.192	0.509 (0.045)
K = 25	Log. Reg.	0.300 (0.063)	0.115 (0.021)	0.166	0.513 (0.027)
	CBA	0.621 (0.121)	0.113 (0.015)	0.191	0.504 (0.027)
K = 50	Log. Reg.	0.236 (0.051)	0.119 (0.018)	0.158	0.505 (0.028)
	CBA	0.612 (0.113)	0.113 (0.012)	0.190	0.513 (0.021)
K = 100	Log. Reg.	0.187 (0.053)	0.117 (0.023)	0.144	0.508 (0.031)
	CBA	0.572 (0.105)	0.110 (0.012)	0.184	0.511 (0.031)
K = 200	Log. Reg.	0.166 (0.049)	<b>0.133</b> (0.025)	0.148	0.515 (0.028)
	CBA	0.390 (0.085)	0.114 (0.018)	0.177	0.506 (0.026)

**Table 1: A table presenting mean value and standard deviation of recall, precision, area under the receiver-operator curve (AROC), and f-measure for both settings: Logistic Regression, CBA. K represents the number of centroids used for the histograms.**

upward jump of precision in Logistic Regression with K=200 explains the increase in f-measure for that method with this single value, denying the general decreasing trend; this was also the highest precision value for this experiment.

These values could be also compared to a baseline of a purely random recommendation. Since there were 533 complete song reproductions out of a recommendation matrix with 4 listeners and 1199 songs, the density of 1's is  $533/(1199 * 4) \approx 0.111$  in the ground-truth, and so generating a uniformly random binary matrix would theoretically produce 0.111 of precision, 0.5 of recall, and an f-measure of 0.182, independently of K. This has also been established by numerical simulations, not reproduced here to save space.

Another baseline which might be interesting to consider is a pure recommendation of 1's (or simply stated "just listen to everything") or a pure recommendation of 0's ("don't listen to anything"). In the first case the theoretical precision would again be 0.111 for this data, and the theoretical recall would be 1.0, with an f-measure of 0.2, whereas the second approach produces recall  $R = 0$  and precision and f-measure are not defined.

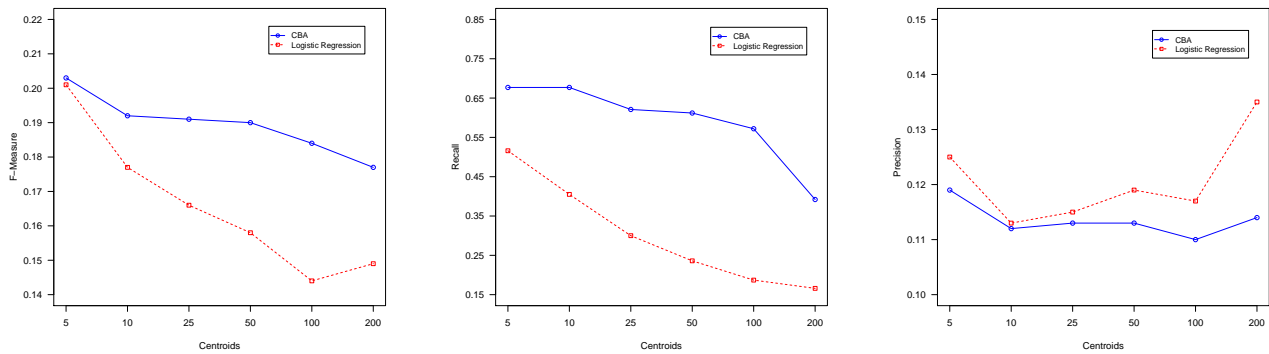
Every setting used in the experiment surpassed the 50% threshold under the ROC curve. This is the curve for defining the configuration of

classifiers in terms of true positive against false positive, with its diagonal meaning random behavior. The highest value achieved was for Logistic Regression operating with 5 centroids.

## 8. Conclusion

CBA has proved to be a good model for predicting sparse listening data for small amounts of MFCC centroids in a Vector Quantized representation. It reached its best f-measure value when MFCC data was represented by only 5 centroids. The hidden  $\beta$  matrix represents the distribution between the centroids and listeners taste, and the results point to the possibility of representing these tastes in 5 dimensions, which may be due to the small number of listeners who participated in the experiment. Repeating the experiment with a larger number of listeners, as well as a more robust method for defining an optimal K value are considered as future work.

Vector-quantization turned out to be expensive in terms of memory consumption for high K values, and this should be also tackled in the future. The cost for running Expectation Maximization for a high number of centroids is also very high, and because of this, good results for low values of K are computationally preferable.



**Figure 3: Comparison between F-measure, recall and precision for CBA and Logistic Regression for all values of K**

## References

- [1] Beth Logan. Music recommendation from song sets. In *In Proc ISMIR*, pages 425–428, 2004.
- [2] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, 2005.
- [3] Jonathan L. Herlocker, Joseph A. Konstan, Al Borchers, and John Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '99*, pages 230–237, New York, NY, USA, 1999. ACM.
- [4] Luis M. de Campos, Juan M. Fernández-Luna, Juan F. Huete, and Miguel A. Rueda-Morales. Combining content-based and collaborative recommendations: A hybrid approach based on bayesian networks. *Int. J. Approx. Reasoning*, 51(7):785–799, September 2010.
- [5] Matthew D. Hoffman, David M. Blei, and Perry R. Cook. Easy as cba: A simple probabilistic model for tagging music. In *10th International Society for Music Information Retrieval Conference*. 2009.
- [6] Kazuyoshi Yoshii, Masataka Goto, Kazunori Komatani, Tetsuya Ogata, and Hiroshi G. Okuno. An efficient hybrid music recommender system using an incrementally trainable probabilistic generative model. In *IEEE Transaction on Audio Speech and Language Processing*, pages 435–447, 2008.
- [7] Wikipedia. Lista dos 100 maiores discos da música brasileira pela Rolling Stone Brasil. [https://pt.wikipedia.org/wiki/Lista\\_dos\\_100\\_maiores\\_discos\\_da\\_musica\\_brasileira\\_pela\\_Rolling\\_Stone\\_Brasil](https://pt.wikipedia.org/wiki/Lista_dos_100_maiores_discos_da_musica_brasileira_pela_Rolling_Stone_Brasil), 2007. [Online; accessed 28-May-2017].
- [8] Ricardo Baeza-Yates, Berthier Ribeiro-Neto, et al. *Modern information retrieval*, volume 463. ACM press New York., 1999.



# A Score-Informed Approach for Pitch Visualisation of a *Cappella* Vocal Quartet Performances

Rodrigo Schramm<sup>1,2</sup>, Helena de Souza Nunes<sup>1</sup>, Emmanouil Benetos<sup>2</sup>

<sup>1</sup>Department of Music, Universidade Federal do Rio Grande do Sul, Brazil

<sup>2</sup>Centre for Digital Music, Queen Mary University of London, UK

rschramm@ufrgs.br, helena.souza.nunes@ufrgs.br, emmanouil.benetos@qmul.ac.uk

## Abstract

This paper presents a score-informed method for visualising the pitch content of polyphonic signals from audio recordings containing *a cappella performances* with multiple singers. A model based on and extending Probabilistic Latent Component Analysis (PLCA) is proposed for estimating the activations of multi-pitch candidates, with the support of a 4-dimensional dictionary built on spectral templates of singer vocalisations. The model is assisted through a soft masking mechanism built from the given music score during the vocal performance. Since the music score is prior knowledge of our system, the main contribution of this method is the potential frame-based visualisation of the fundamental frequencies of each vocal part, which can be further used for singing analysis including tuning, vibrato and portamento analysis. We evaluate our system on recordings of vocal quartets, including Bach Chorale and Barbershop styles. The evaluation process also takes into account possible discrepancies between the singing performance and the original music score. Experimental results show the influence of such mismatches on the final system accuracy.

## 1. Introduction

The visualisation of multi-pitch content (also known as pitch salience) of polyphonic music generated by multiple singers is useful information for singing analysis of tuning, vibrato, portamento and a variety of complex pitch contours. Often, multi-pitch salience is an intermediate step of automatic music transcription algorithms, which convert audio signals into a symbolic representation (such as a music score) and can further be used to support applications in mu-

sic information problems, computational musicology, interactive music systems, and automatic music assessment.

A method for visualising the pitch content of polyphonic music signals was proposed in [1], where a pitch salience function was designed to produce continuous pitch values. With a similar aim, [2] presents an approach using the Fan Chirp Transform [3] for pitch visualisation. Despite the interesting results obtained in these two approaches, a robust method for visualisation of the multi-pitch content of polyphonic signals is still needed. In fact, unsupervised techniques without any additional prior information usually contain many errors since the mixture of the harmonic content from different sung notes tends to generate false positives.

Spectrogram factorisation algorithms have been extensively applied for automatic music transcription and source separation in the last decade, including approaches as non-negative matrix factorization (NMF) and probabilistic latent component analysis (PLCA) [4–7]. In these approaches, the input time-frequency representation (spectrogram) is decomposed into non-negative factors, consisting mainly of spectrum atoms and note activations. Techniques based on spectrogram factorisation have shown a straightforward framework for score-informed approaches [8, 9] since masking can be applied to the matrix's coefficients, guiding the convergence of the optimisation algorithm.

In this paper, we propose a frame-based system for visualising the pitch content from polyphonic audio recordings. Our system uses a variation of the spectrogram factorization method described in [10], and its scope focuses on audio recordings of *a cappella* performance with multi-

ple singers. The original method has shown good results for (blind) multi-pitch detection. However, it is not able to perform voice separation, i.e. assign each detected note to a specific voice type (e.g. soprano). In our new approach, we overcome the voice separation limitation by integrating the music score<sup>1</sup> information through a soft masking scheme. Thus, since the music score is considered prior knowledge, the central point of this new application is neither on pitch detection nor note transcription, but on the detailed visualisation of the pitch contour of each vocal part. Figure 1 shows an example of the output generated by our system. The top image in this figure shows the spectrogram estimated using the Variable-Q Transform representation [11] with 20 cent frequency resolution and frame with hop size of 20 ms. In the middle is shown the ground truth, where each colour means a vocal part (SATB), and on the bottom is shown the estimated multi-pitch visualisation obtained through the proposed score-informed model.

The remainder of this paper is organised as follows. Section 2 describes the proposed score-informed PLCA model with the soft masking procedure. Section 3 presents the experiments used to evaluate the system accuracy, including simulations of singing mistakes in order to quantify the impact caused by discrepancies between the notes that are out of the soft mask range. Section 4 draws conclusions and future work.

## 2. Proposed Method

Our system for multi-pitch visualisation is a simplified variant of the spectrogram factorisation process described in [10]. In this model, the input time-frequency signal representation is decomposed into several components denoting the activations of pitches, voice types, and singer-timber atoms. This factorisation is supported by the use of a fixed dictionary of log-spectral templates, which are extracted from solo singing recordings in the RWC audio dataset [12]. In order to build the dictionary, we used recordings from subjects of distinct voice types: bass, baritone, tenor, alto, soprano. The dictionary has spectral templates from 15 distinct singers

<sup>1</sup>We use a MIDI representation in the case of this work.

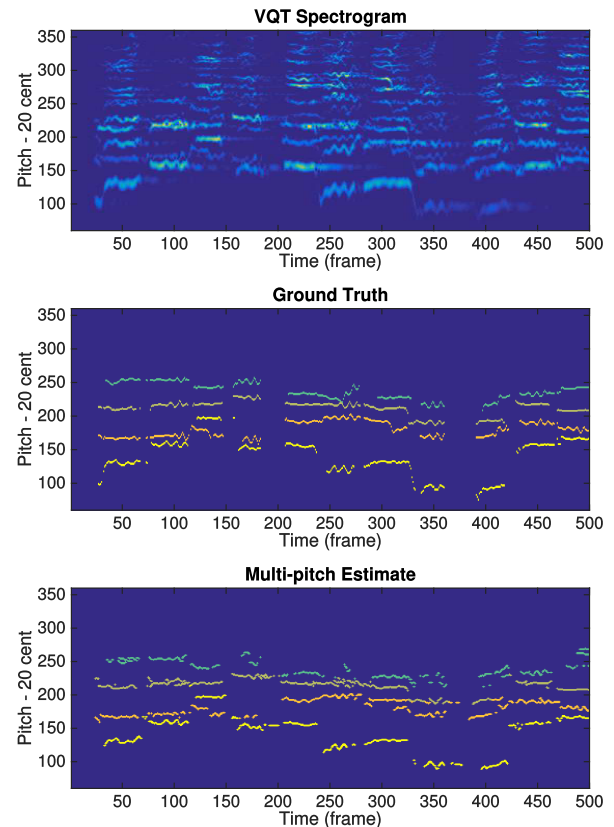


Figure 1: Multi-pitch visualization

(9 male and 6 female), that have sung sequences of notes following a chromatic scale and distinct vowels (/a/, /æ/, /i/, /o/, /u/).

This collection of pre-extracted spectral templates is represented by  $\Phi = P(\omega|s, p, v)$ , where variable  $p \in \{105, \dots, 540\}$  denotes pitch in log-frequency scale (12-tone equal temperament in 20 cent resolution scale from MIDI pitch 21 to 88),  $s$  denotes the singer-timber atom index (15 distinct singers),  $v$  denotes the voice type (e.g. soprano, alto, tenor, bass). Both the input signal and the spectral templates use a normalised variable-Q transform (VQT) representation [13]. Details on the procedure for the construction of a similar dictionary are available in [10].

### 2.1. Multi-pitch estimation

The input VQT spectrogram is denoted as  $X_{\omega, t} \in \mathbb{R}^{\Omega \times T}$ , where  $\omega$  denotes log-frequency and  $t$  time. In the model,  $X_{\omega, t}$  is approximated by a bivariate probability distribution  $P(\omega, t)$ , which is in turn decomposed as:

$$P(\omega, t) = \sum_{s, p, v} P(t) \Phi P_t(s|p) P(v) P_t(p|v) \quad (1)$$

where  $P(t)$  is the spectrogram energy (known quantity).

This model decomposes the probabilities of pitch and voice type as  $P(v)P_t(p|v)$ .  $P(v)$  is the mixture weight that denotes the overall contribution of each voice type presenting in the input recording and  $P_t(p|v)$  denotes the pitch activation for a specific voice type (eg. SATB) over time. The contribution of specific singer subjects from the training dictionary is modelled by  $P_t(s|p)$ , i.e. the singer-timber contribution per pitch over time. All unknown model parameters  $P_t(s|p)$ ,  $P_t(p|v)$ , and  $P(v)$  are estimated through the iterative expectation-maximization (EM) algorithm [14].

In the *Expectation* step we compute the posterior as:

$$P_t(s, p, v|\omega) = \frac{\Phi P_t(s|p) P(v) P_t(p|v)}{\sum_{s,p,v} \Phi P_t(s|p) P(v) P_t(p|v)} \quad (2)$$

In the *Maximization* step, each unknown model parameter is then updated by:

$$P_t(s|p) \propto \sum_{v,\omega} P_t(s, p, v|\omega) X_{\omega,t} \quad (3)$$

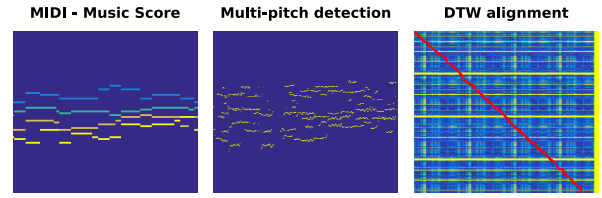
$$P_t(p|v) \propto \sum_{s,\omega} P_t(s, p, v|\omega) X_{\omega,t} \quad (4)$$

$$P(v) \propto \sum_{s,p,\omega,t} P_t(s, p, v|\omega) X_{\omega,t} \quad (5)$$

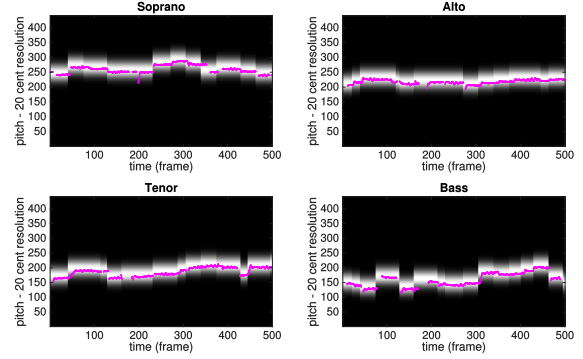
The model parameters are randomly initialised, and the EM algorithm iterates over Eqns (2)-(5). In our experiments we use 25 iterations. The output of the PLCA model is a 20 cent-resolution time-pitch representation for each voice type, given by  $P(p, v, t) = P(t)P(v)P_t(p|v)$ .

## 2.2. Soft Masking

This multi-pitch estimation model without any additional information does not perform well the voice separation. Aiming to overcome this drawback we introduce a soft masking mechanism. The soft mask is generated from the music score which was used as the reference for the singing performance.



(a)



(b)

**Figure 2: Soft mask generation: a) multi-pitch detection and score alignment; b) generated soft mask per vocal part.**

For automatically aligning the reference MIDI score with the audio recording made by the vocal quartet, we employ a dynamic time warping (DTW) algorithm [15]. Throughout this process, each note from the music score is time-aligned with the sung notes present in the audio recording, such that an optimal match between two given sequences (multi-pitch detection over time and the frame-based representation of the MIDI score) is found.

The proposed DTW algorithm in this paper uses a particular local cost function:

$$C(\mathbf{m}_{t_i}, \mathbf{h}_{t_j}) = \min\left(\sum_{p_v \in \{\mathbf{h}_{t_j}\}} \min(|p_v - \mathbf{m}_{t_i}|^2), \beta\right), \quad (6)$$

for measuring the distance between the list of multi-pitch estimates  $\mathbf{m}_{t_i}$  at frame time  $t_i$  and the list of notes  $\mathbf{h}_{t_j}$  from the music score at frame time  $t_j$ .  $\beta$  is a constant that imposes a limit in the cost contribution when there is no good match between points  $t_i$  and  $t_j$ .

After the time alignment the notes from the music score, at each time frame  $t$ , are used to generate the soft mask such that

$$M_t(p|v) \sim \mathcal{N}(p_t^v, \sigma_m) \quad (7)$$

follows a normal distribution  $\mathcal{N}$  centred at pitch  $p_i^v$ , from the music note  $p$  at voice  $v$ , and with standard deviation  $\sigma_m = 20$  bins (equivalent to 4 semitone). The soft mask is normalised along the frequency bins. Figure 2 illustrates this process.

### 2.3. Joint multi-pitch visualisation and voice separation

The PLCA model is initialised with random parameters, without using the soft mask scheme. The algorithm iterates until the model convergence (usually after 15 iterations). At this stage, we extract the multi-pitch detection and perform the score time alignment to generate the soft mask. After this point, the spectrogram factorisation continues over Eqns (2)-(5). However, Eqn (4) is replaced by

$$P_t(p|v) \propto \alpha \left( \sum_{s,\omega} P_t(s,p,v|\omega) X_{\omega,t} \right) + (1 - \alpha) \phi_t(p|v) \quad (8)$$

where  $\alpha$  is a weight parameter controlling the effect of the soft mask (we have set  $\alpha = 0.5$  based on our experiments) and  $\phi$  is a hyperparameter defined as:

$$\phi_t(p|v) \propto M_t(p|v) P_t(p|v). \quad (9)$$

The hyperparameter of Eqn (9) acts as a soft mask, reweighing the pitch contribution of each voice regarding only the pitch neighbourhood previously defined by the aligned notes in the music score.

## 3. Evaluation

The proposed multi-pitch visualisation system is evaluated on two datasets of *a capella* recordings<sup>2</sup>. These datasets contain audio recordings of 26 Bach Chorales and 22 Barbershop quartets, respectively. All files are in four-channel wave format with a sample rate of 22.05 kHz and 16 bits per sample. Each channel corresponds to a particular vocal part

<sup>2</sup>Original recordings available at <http://pgmusic.com>.

(SATB). The Barbershop dataset contains only male voices, while the Bach Chorale dataset contains a mixture of two male and two female voices. We have extracted a frame-based pitch ground truth for each vocal part by using a monophonic pitch tracking algorithm [16] on each monophonic track. Experiments are conducted using the mix down of each audio file (i.e. polyphonic content), not the individual tracks.

We evaluate the multi-pitch visualisation and the respective voice separation capabilities of the proposed system by using metrics commonly used for multi-pitch detection and automatic transcription evaluation [7]. In these experiments, we estimate the frame-based precision, recall and F-measure as defined in the MIREX multiple-F0 estimation evaluations [17], with a frame hop size of 20 ms. For this, we use the individual voice ground truths and define voice-specific F-measures of  $F_s$ ,  $F_a$ ,  $F_t$ , and  $F_b$  for each respective SATB vocal part. We also define an overall voice assignment F-measure  $F_{va}$  for a given recording as the arithmetic mean of its four voice-specific F-measures.

### 3.1. Results

Our system generates joint multi-pitch visualisation and voice separation. For comparison with other benchmark techniques, which are originally measured in semitone scale, we have down-sampled our system output into 88 MIDI semitones. The joint multi-pitch detection (after the binarization of the pitch activations) and voice separation output is named as MASK4-VA and MASK4-VA-20 for the semitone representation and for the 20 cent resolution, respectively.

Table 1 shows the F-measure comparisons between our proposed method and other three baseline techniques: MSINGERS-VA [10], VOCAL4-MP, and VOCAL4-VA [18]. All the benchmark techniques are PLCA-based models, but they are not score informed approaches. In addition, VOCAL4-VA also implements a language model based on Hidden Markov models to improve the voice separation results.

From the multi-pitch detection results shown in Table 1, it can be seen that MASK4-VA achieves very high  $F_{va}$  on both datasets. This

Model	Bach Chorales				
	$F_{va}$	$F_s$	$F_a$	$F_t$	$F_b$
MSINGERS-VA	18.02	15.37	17.59	26.32	12.81
VOCAL4-MP	21.84	12.99	10.27	22.72	41.37
VOCAL4-VA	45.31	26.07	37.63	49.61	67.94
MASK4-VA	82.93	72.16	80.22	90.65	88.70
MASK4-VA-20	55.93	56.59	53.50	58.14	55.48
Model	Barbershop Quartets				
	$F_{va}$	$F_s$	$F_a$	$F_t$	$F_b$
MSINGERS-VA	12.29	9.70	14.03	27.93	9.48
VOCAL4-MP	18.35	2.40	10.56	16.61	43.85
VOCAL4-VA	46.92	40.01	35.57	29.76	82.34
MASK4-VA	78.75	69.14	71.97	88.87	85.02
MASK4-VA-20	51.31	47.16	50.59	57.77	49.75

Table 1: Voice assignment results.

good performance is already expected since our approach is score-informed. The  $F_{va}$  measure for the MASK4-VA-20 (20 cent resolution) is substantially lower. This mainly occurs because of small discrepancies between the ground truth and the pitch tracking in regions with vibrato.

Another important evaluation is done regarding the vulnerability of our system when there is the presence of singing mistakes, i.e., when the sung notes mismatch the target notes in the music score. To simulate this situation, we randomly change the pitch value of a percentage of notes from the music score. The plot in the Figure 3 shows the F-measure evolution as the percentage of wrong notes increases for the Bach Chorales dataset. The decrease in accuracy is caused by two main factors: 1) direct mismatch between the ground truth and the sung note; 2) incorrect time alignment from the DTW. The second factor is a consequence of the first. This result implies that our technique is more suitable for singing recordings that are reliable performances of the respective reference music scores.

#### 4. Conclusion

In this paper, we have presented a score-informed method for visualising the pitch content of polyphonic signals from audio recordings containing a cappella performances with multiple singers. The proposed system uses a spectrogram factorisation model for multi-pitch detection and a soft mask scheme for aiding voice assignment. We have evaluated our system on two datasets (Bach Chorales and Barbershop styles),

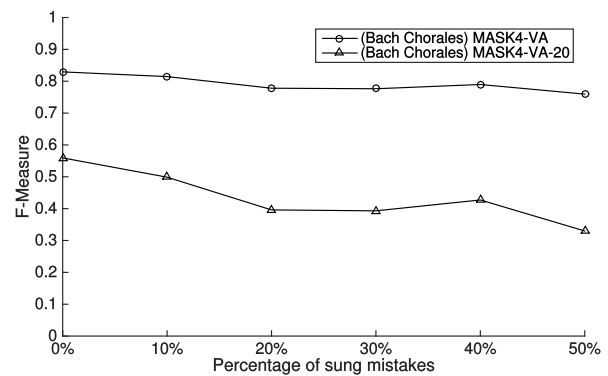


Figure 3: F-measure evolution as a function of sung mistakes.

comparing results with baseline approaches for multi-pitch detection and voice assignment.

Experimental results have shown that the soft-masking scheme improved the multi-pitch visualisation, ensuring good voice assignment. However, our system is vulnerable to singing mistakes since the soft-mask depends on the alignment between the singing performance and the reference music score. Thus, there is certainly room for improvement. Avenues for future work include a better handling of singing mistakes during the music score alignment and the search for alternative and robust masking approaches.

#### 5. Acknowledgement

RS is supported by a UK Newton Research Collaboration Programme Award (grant no. NRCP1617/5/46). EB is supported by a UK Royal Academy of Engineering Research Fellowship (grant no. RF/128).

#### References

- [1] Anssi Klapuri. A method for visualizing the pitch content of polyphonic music signals. In *Proceedings of the 10th International Society for Music Information Retrieval Conference, ISMIR 2009, Kobe International Conference Center, Kobe, Japan, October 26-30, 2009*, pages 615–620, 2009.
- [2] Luis Jure, Ernesto López, Martín Rocamora, Pablo Cancela, Haldo Sponton, and Ignacio Irigaray. Pitch content visualization tools for music performance anal-

- ysis. In *Proceedings of the 13th International Society for Music Information Retrieval Conference, ISMIR 2012, Mosteiro S.Bento Da Vitória, Porto, Portugal, October 8-12, 2012*, pages 493–498, 2012.
- [3] Martín Rocamora Pablo Cancela, Ernesto López. Fan chirp transform for music representation. In *In Proceedings of the 13th International Conference on Digital Audio Effects (DAFx-10)*. Graz, Austria, pages 330–337, 2010.
- [4] Shrikant Venkataramani, Nagesh Nayak, Preeti Rao, and Rajbabu Velmurugan. Vocal separation using singer-vowel priors obtained from polyphonic audio. In *Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR 2014, Taipei, Taiwan, October 27-31, 2014*, pages 283–288, 2014.
- [5] Gautham J. Mysore and Paris Smaragdis. Relative pitch estimation of multiple instruments. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2009, 19-24 April 2009, Taipei, Taiwan*, pages 313–316, 2009.
- [6] G. Grindlay and D. P. W. Ellis. Transcribing multi-instrument polyphonic music with hierarchical eigeninstruments. *IEEE Journal of Selected Topics in Signal Processing*, 5(6):1159–1169, Oct 2011.
- [7] Emmanouil Benetos and Tillman Weyde. An efficient temporally-constrained probabilistic model for multiple-instrument music transcription. In *Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR 2015, Málaga, Spain, October 26-30, 2015*, pages 701–707, 2015.
- [8] Emmanouil Benetos, Anssi Klapuri, and Simon Dixon. Score-informed transcription for automatic piano tutoring. In *Proceedings of the 20th European Signal Processing Conference, EUSIPCO 2012, Bucharest, Romania, August 27-31, 2012*, pages 2153–2157, 2012.
- [9] S. Ewert, B. Pardo, M. Muller, and M. D. Plumbley. Score-informed source separation for musical audio recordings: An overview. *IEEE Signal Processing Magazine*, 31(3):116–124, May 2014.
- [10] R. Schramm and E. Benetos. Automatic transcription of a cappella recordings from multiple singers. In *AES International Conference on Semantic Audio*, June 2017.
- [11] C. Schörkhuber and A. Klapuri. Constant-q transform toolbox for music processing. In X. Serra, editor, *Proceedings of 7th Sound and Music Computing Conference*, Barcelona (Spanien), 12 2010. procedure: peer-reviewed.
- [12] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC music database: Music genre database and musical instrument sound database. In *ISMIR*, pages 229–230, 2004.
- [13] C. Schörkhuber, A. Klapuri, N. Holighaus, and M. Dörfler. A Matlab toolbox for efficient perfect reconstruction time-frequency transforms with log-frequency resolution. In *AES 53rd Conference on Semantic Audio*, January 2014.
- [14] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, 39(1):1–38, 1977.
- [15] Meinard Müller. *Information Retrieval for Music and Motion*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.
- [16] Matthias Mauch and Simon Dixon. pyin: A fundamental frequency estimator using probabilistic threshold distributions. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2014)*, 2014.
- [17] M. Bay, A. F. Ehmann, and J. S. Downie. Evaluation of multiple-F0 estimation and tracking systems. In *ISMIR*, pages 315–320, October 26-30 2009.
- [18] Rodrigo Schramm, Andrew McLeod, Mark Steedman, and Emmanouil Benetos. Multi-pitch detection and voice assignment for a cappella recordings of multiple singers. In *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China, October 23-27, 2017*, 2017.

# aAaA: an attribute aware abstraction architecture allowing arbitrary argument assignment in Pure Data

José Henrique Padovani<sup>1\*</sup>

<sup>1</sup>NICS - Interdisciplinary Nucleus of Sound Communication / State University Of Campinas (UNICAMP)  
Music Department / Arts Institute / State University Of Campinas (UNICAMP)  
R. Elis Regina, 50 – Cidade Universitária – Campinas, SP – CEP 13083-854

josep@unicamp.br

## Abstract

We describe aAaA, an abstraction-based extension for Pure Data (Pd) that parses any number of attributes (@-initiated symbols) and their associated arguments and routes the entered values to unique labeled receivers. This approach expands the syntax of Pd without the need of compiled libraries, objects and extensions – a feature that can be useful in contexts in which Pd abstractions are embedded in applications, mobile and similar architectures.

## 1. Introduction

In the last decades Pure Data (Pd) has become a major music and audio programming environment, being extensively used in artistic contexts, pedagogical situations, entertainment products and research projects. Due to its intuitive graphical coding approach (consisting of interconnecting objects, messages and other data structures to route symbolic control informations and digital audio signals) and also to its multi-platform free and open source software licensing, compatibility and distribution, Pd has spread rapidly as a flexible and powerful tool to process and synthesize sounds in real-time [1].

Recently, it also became a very convenient audio framework, being possible to be embedded in other applications and run not only in conventional desktop and laptop computers but, also, in mobile platforms and single-board computers[2, 3]. In these environments, it is usually easier to run Pd “patches” that do not require any compiled extensions. If this is sometimes seen as a limitation – specially taking into account that the

increasing use of Pd was largely due to the now discontinued “Pd-extended” fork, that was distributed with a large set of compiled extensions –, it is possible to make Pd abstractions that reproduce the features provided by these extended objects and functions. This can be done with coding/‘patching’ strategies such as *dynamical patching*, without causing any significant loss of performance<sup>1</sup>.

Taking this context into account, this paper presents an abstraction extension entirely created with Pd vanilla resources that expands the argument syntax of Pd abstractions entirely written as patches that use only usual vanilla objects. Developed to be run in Pd vanilla, aAaA can be seen as a template abstraction that emulates some of the Max<sup>2</sup> features related to how it deals with objects and their arguments. In some Max objects, it is possible to assign arguments by entering one or more attributes and related values. A Max [cycle~] oscillator object, for instance, can be created with arguments such as @phase, @frequency, @buffer and other attribute tags followed by their respective values (a handy syntax feature that makes patching

<sup>1</sup>Until few years ago, Pd was distributed in two major versions. Pd “vanilla” is the main branch of development, being mostly carried by Miller Puckette with few core objects. The “extended” fork was distributed in a larger package with many compiled extensions and objects. It was mainly maintained by Hans-Christoph Steiner and is now not supported/developed any more. To compensate this, Pd-vanilla has, since the version 0.47.0, its own extensions install manager (deken). It is also worth to mention a new cross-platform Pd branch: Purr Data/Pd-l2ork, which comes with many pre-compiled extensions and features a modern HTML5 based GUI. More information about Pd “flavors” can be reached at <https://puredata.info/>

<sup>2</sup> <http://cycling74.com/products/max>.

\*Supported by CNPq [Edital Universal 14/2014] and FAEPEX/UNICAMP



more user-friendly/mnemonic and makes possible to enter only the relevant non-default values desired in a specific context)[4].

While similar functionalities have been already implemented in compiled objects and libraries in Pd (as it happens, for example, with the library `cyclone`, designed to clone many Max features in Pd)[5] and while other Pd compiled objects use “flags” to set specific internal parameters and values, `aAaA` implements this same kind of syntax recognition inside Pd by taking advantage of dynamical patching processes.

## 2. Pd abstractions and arguments

A parsing process that iterates given attributes and arguments of an abstraction is not trivial to be constructed with Pd patches. Indeed, as Pd is a graphical language, processes like iterative loops or parsing processes are constructed in a very different way than similar processes programmed in functional or object-oriented code based languages. Particularly regarding argument parsing, if it is desired that some abstraction uses arguments that are informed in the moment of the object instantiation, it is necessary, in order to use these values to some purpose, to know exactly how many arguments will be used and in which order they will be informed. In this way, one can use Pd objects and *dollarsign* arguments to retrieve the values informed by the user/developer when he instantiates the object. If the abstraction patch has a `[float $1]` and a `[symbol $2]` inside, these two objects will store respectively a number and a string that can be then used internally to change parameters of the algorithm.

Briefly, the `aAaA` algorithm parses an arbitrary number of symbols and floats given as the objects arguments, and groups them in lists for each *attribute* – a pseudo-type that consist in a symbol/string that begins with the `@` character. These lists are used internally to route the attribute-nestled parameters to the appropriate receivers (`[receive]`/`[r]` objects). Arguments given before any `@`-initiated symbol are internally grouped in a single list to be sent to its respective receiver object, being possible to split

them in individual symbols, *unpack* or subject them to other parsing/processing mechanisms.

In the following sections, further details are giving explaining each step of the algorithm and exemplifying the application of `aAaA` in a simple synthesis abstraction based object.

## 3. aAaA: any Arguments and Attributes

Internally, `aAaA` has a ‘subpatch’ (a container where a sub-process runs) named `[pd aAaA-kernel]` (Fig. 1). This subpatch encapsulates the mechanism that parses arguments such as floats, symbols and attributes and sends them to the appropriate `[receive]` objects that may used those values if they are informed by the user/developer.

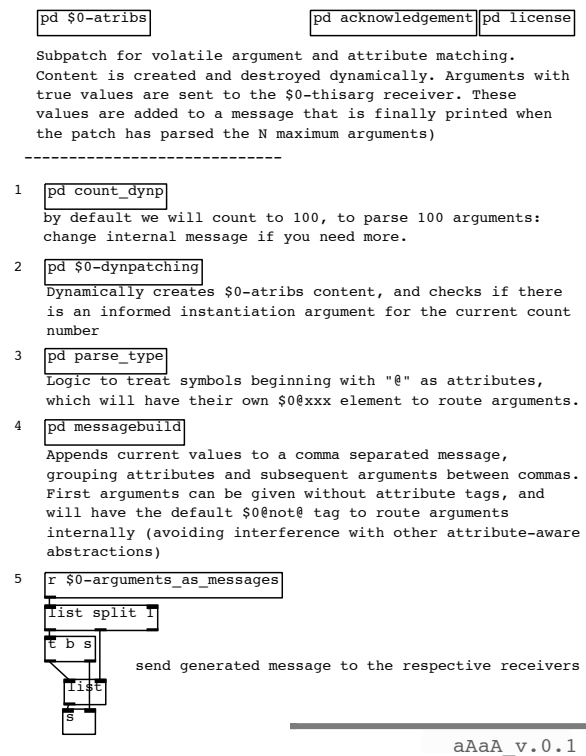


Figure 1: `[pd aAaA-kernel]` subpatch content

The `[pd aAaA-kernel]` ‘subpatch’ is designed be copied to any Pd abstraction being written, enabling it to parse attributes/arguments internally.

`[pd $0-atribs]` (Fig. 2) is a blank container whose content will be iteratively populated and destroyed with dynamically created objects.



This is done to retrieve each of the arguments given in the instantiation of the aAaA-enabled abstraction, according to a counter that is driven in the [pd count\_dynp] sub-patch (fig. 2). This patch controls the main counter, and send ordered messages to initiate the whole process, to execute the dynamical patching mechanism and to inform which is the current argument index from 1 to N (that has the default value of 100, which can be changed for specific situations).

Main Counter and dynamic patcher control (counts until N to parse input arguments)

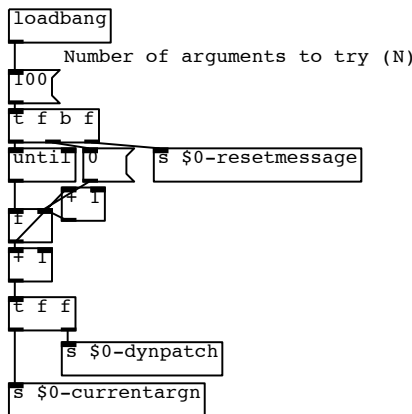


Figure 2: [pd count\_dynp] content

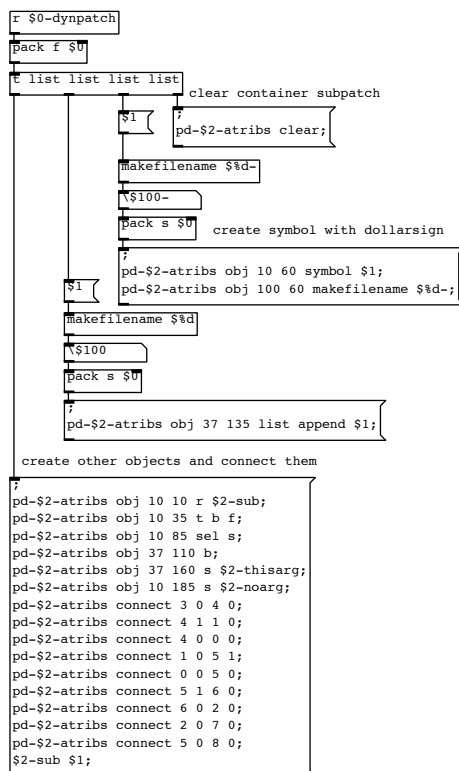


Figure 3: [pd \$0-dynpatching] content

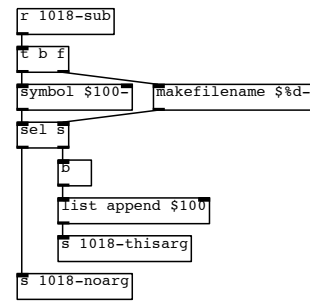


Figure 4: The dynamically created [pd \$0-attrs] subpatch content at the moment that the counter has reached the number 100

The [pd \$0-dynpatching] (fig. 3) process creates and deletes dynamically the content of the abstraction [pd \$0-attrs] (fig. 4) and retrieves the argument in the current index informed by the main counter, checking if it really was informed in the moment of object instantiation.

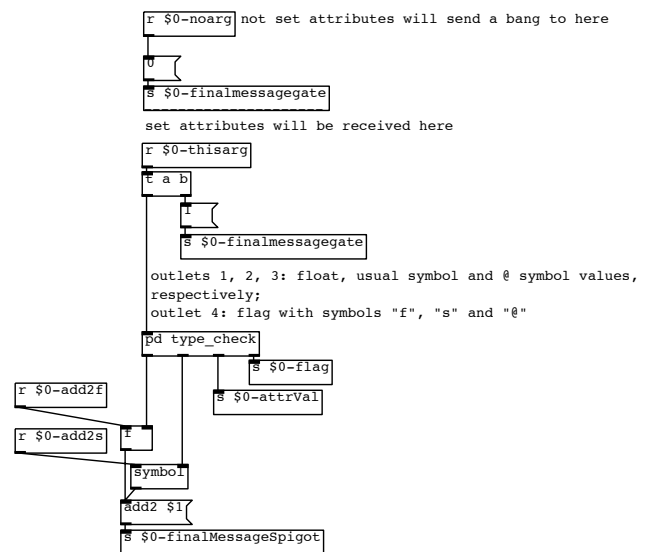


Figure 5: The [pd parse\_type] content.

The [pd parse\_type] (fig. 5) sub-patch checks if the argument is a floating point number (f), a usual symbol/string (s) or an attribute (@) – i.e., a symbol that begins with the char @. The flags f, s and @ are sent to the following processes, being used to organize a growing list of comma separated messages that will be sent at the end of the parsing process to local receivers named according to the attribute tags.

The [pd messagebuild] (fig. 6) process organizes a big comma separated list, that will route attribute-grouped arguments together, sending then to a final receiver – see stage 5. in the fig. 1. This final receiver routes the individual messages to specific receivers that will be named according to the attributes, preceded by the unique abstraction wildcard \$0, which is instantiated locally for each abstraction in Pd. This enables the use of multiple aAaA-enabled abstractions side-by-side, and allows to create scalable, flexible and structured objects capable of dealing with the instantiation arguments in a very intuitive way.



Figure 6: The [pd messagebuild] sub-patch content.

#### 4. Using aAaA: an example

aAaA is distributed with two abstractions. aAaA-example.pd (fig. 7) uses the described [pd aAaA-kernel] (fig. 1) mechanisms to create a table lookup synthesis process with two amplitude modulation oscillators; and aAaA-help.pd which instantiates the previous abstraction with a series of attributes and

arguments that are internally assigned to the respective objects and data structures that are used in the synthesis process.

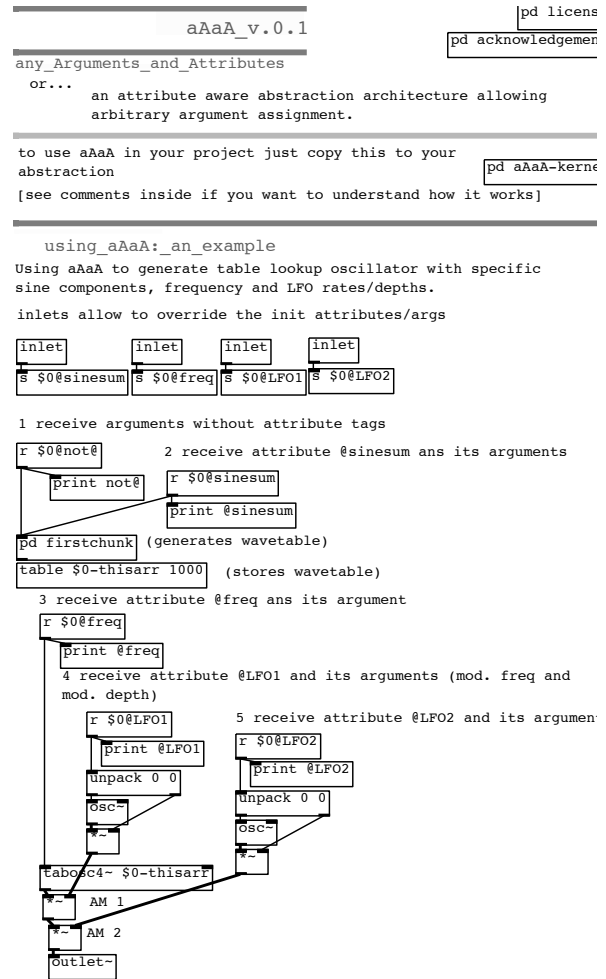


Figure 7: The aAaA-example.pd abstraction, using the aAaA kernel to parse parameters of a table lookup synthesis process.

#### 4.1. aAaA-example.pd

The aAaA-example.pd (fig. 7) abstraction uses the messages sent by the [pd aAaA-kernel] to control different parameters of the synthesis algorithm. The first attribute is @sinesum, with arguments that are also programmed to be received without any attribute tag. These arguments are internally received by objects [r \$0@sinesum] and [r \$0@not@] (a special tag used for arguments without any attribute specification). They are used in the sub-patch [pd firstchunk] to populate the table \$0-thisarr\$, that will

be continuously read by a quadratic interpolating table oscillator ([`tabosc4~`]) with a frequency given by a floating number preceded by the `@freq` attribute tag (and that will be routed to the [`r $0@freq`] object).

The attributes `@LFO1` and `@LFO2` expect, each one, two floating point arguments that control, respectively, the modulation frequency and the modulation depth of two amplitude modulation processes. These values are received by the [`r $0@LFO1`] and the [`r $0@LFO2`] objects, being unpack and appropriately routed internally to drive the modulation oscillators.

## 4.2. aAaA-help.pd

The `aAaA-help.pd` abstraction (fig. 1) exemplifies how objects created with the `aAaA-kernel` attribute and argument parsing mechanism can take advantage of the `aAaA` attribute syntax.

Given that the `aAaA-example.pd` and the `aAaA-kernel` processes use `$0` wildcards – which correspond to individual numbers that identify each abstraction instance – to name objects, tables and data structures, `aAaA`-enabled abstractions don't interfere in each other. In other words, if two objects share the same attribute tags names (in the given example, for instance, the attribute `@LFO1`), only the internal receiver identified by the individual number of `@0` followed by the attribute symbol `@LFO1` (thus, the receiver `$0@LFO1`) will receive the arguments intended to be used internally.

## 5. Conclusion

`aAaA` is capable of expanding the coding syntax of Pd without requiring any special libraries and extensions: a feature that allows, among other things, that `aAaA`-enabled abstractions to be easily embedded in applications and hardwares that use `libpd` as an audio processing/synthesis framework. As the dynamical process of parsing attributes and arguments occurs in the instantiation stage of the abstractions, the use of `aAaA` does not imply in any loss of performance.

The same strategy to route `@` initiated symbols in `aAaA` could be used to recognize other pseudo-types. It would be possible, for instance, to use the commonly used hyphen symbol (`-`) to designate flags that should be treated in a special way. Likewise, one could designate a set of symbols to instantiate special data structures like matrices or trees.

`aAaA` is part of a larger set of abstractions, objects and resources currently being developed in the Interdisciplinary Nucleus of Sound Communication (NICS). This set of resources are aimed to expand the coding possibilities of Pd and are designed to be easily adapted to specific circumstances and to be able to be embedded in heterogeneous platforms, applications, hardwares and operational systems. In a more general perspective, the objective is to develop resources to design structured abstractions whose mechanisms and general behavior can be specified in a more structured way, just like code based computer music languages usually feature.

## 6. Acknowledgments

`aAaA` is based in previous abstractions and strategies developed by IOhannes m zmölnig that were distributed in the Pd mail-list<sup>3</sup>.

## References

- [1] Miller Puckette. Etude de cas sur les logiciels pour artistes: Max/MSP et Pure Data. In David-Olivier Lartigaud, editor, *Art++*. HYG, Orléans, July 2011.
- [2] Peter Brinkmann, Dan Wilcox, Tal Kirshboim, Richard Eakin, and Ryan Alexander. Libpd: Past, Present, and Future of Embedding Pure Data. *PdCon2016~*, 2016.
- [3] Peter Brinkmann. *Making Musical Apps: Real-Time Audio Synthesis on Android and iOS*. O'Reilly Media, Sebastopol, Calif, 1 edition edition, March 2012.

<sup>3</sup>Relevant discussions can be found at the following archive links: <https://lists.puredata.info/pipermail/pd-list/2008-10/065465.html> and <https://lists.puredata.info/pipermail/pd-list/2016-08/115936.html>.

- [4] MSP reference - cycle~ - sinusoidal oscillator. <https://docs.cycling74.com/max7/maxobject/cycle~>.
- [5] Alexandre Torres Porres, Derek Kwan, and Mathew Barber. Cloning Max/MSP Objects: A Proposal for the Upgrade of Cyclone. *Pd-Con2016~*, 2016.

# Challenges for a Second Decade of Ubimus Research: Metaphors for Creative Action (part 1)

Damián Keller<sup>1\*</sup>

<sup>1</sup>Amazon Center for Music Research (NAP) – Federal University of Acre and  
Federal Institute of Acre, Rio Branco, Acre, Brazil

dkeller@ccrma.stanford.edu

## Abstract

This is the first part of a discussion on the challenges of a second decade of ubimus research. I lay out and exemplify the concept of metaphor for creative action. I summarize the results of three studies employing the time tagging metaphor, configuring an effective strategy for supporting everyday musical creativity. Then I report results of a study employing the stripe metaphor – an extension of time tagging devised for usage of a large number of resources. Twelve subjects, encompassing musicians and casual participants, realized improvisatory sessions in a non-standard setting – an audio and musical equipment store. The results indicated a promising new avenue of research targeting lay-musician interaction.

## 1. Introduction

In 2009, after two years of intense exchange – led by two Brazilian research groups, NAP (UFAC/IFAC) and LCM (UFRGS) – the initial proposal on ubiquitous music research (ubimus) was laid out in a series of papers and artworks presented at the Congress of the ANPPOM [1] and at the Brazilian Symposium on Computer Music [2, 3]. Subsequently, case studies and artistic products were presented as invited exhibits, talks and panels at the Biennial of Latin American Art in Denver, Colorado (2013), ANPPOM (2014), SIMA (2015) and SEMPEN (2016). An upcoming issue of *Per Musi* features a section dedicated to ubimus and special volumes were published by Sonic Ideas (2013), *Cadernos de Informática* (2014) and *Scientia*

Tech (2015). Aside from the multiple chapters and papers that have appeared in specialized publications over the last few years – such as the *Journal of New Music Research*, *Organised Sound*, and the *Journal of Music Technology and Education* – a reference volume was released by Springer Press in 2014. Hence, I believe we can say that ubiquitous music constitutes a consolidated research field.

One of the objectives of ubimus endeavors is to provide access to creative music making for a wide range of participants. Supporting good-quality musical products without creating unnecessary barriers to novice participation is particularly tricky. Hence, ubimus research has yielded alternative approaches, including the development of creativity support metaphors. These metaphors can be used to guide the implementation of technological infrastructure. Whether the metaphors are effective means of support for creative activities demands experimentation and data collection in real settings. Thus, ubimus studies deal with the assessment of creative products and processes while subjects carry out musical activities in everyday contexts.

In this paper, I summarize and discuss the results of field studies employing time tagging [4] and report results of a study employing the stripe metaphor [5]. Firstly, I lay out the concept of metaphor for creative action. This is intrinsically linked to ubimus research, and to the best of my knowledge, it has not been articulated in related fields – including computer music, creativity studies or human-computer interaction. The second section of the paper provides a short description of the time tagging metaphor and summarizes the experimental findings of three ubimus studies, highlighting the limitations of the initial implementations. Part of these limitations are

\*Research partially supported by a CNPq Productivity Grant 2015-2017. Special thanks to Edemilson Ferreira for his collaboration in this project.

addressed by the second generation of mixDroid prototypes, embodying the stripe metaphor. Access to massive resources is one of the factors considered in this new set of support tools. I report a study involving both musically trained and lay subjects in everyday settings. Lay-musician interaction emerges as one of the key findings of the experiment. Whether the phenomenon of increased engagement is linked to the participation of both musicians and naive subjects is an experimental question that demands further study. In the last section of the paper, I place these issues within the larger context of future research endeavors in ubimus.

## 2. Metaphors for creative action

Creativity support metaphors furnish a contact point between musical interaction metaphors [6] and the approaches to creativity laid out in interaction aesthetics [7]. Metaphors for creative action differ from domain-specific musical interaction metaphors (see the proceedings of the NIME conferences for multiple examples of the latter). While musical interaction metaphors strive to provide support for musicians within the context of executive activities, metaphors for creative action strive to increase the participants' creative potentials. Hence, they are applicable to a variety of design activities, including planning and exploration. Creative potentials can impact the intended and the unintended by-products of the activity. So they not only target explicit cognitive processes. Metaphors for creative action may find application when the activity demands usage of tacit knowledge.

The creativity support metaphors described in the following sections – time tagging and the stripe metaphor – employ designs based on ecologically grounded strategies [4]. Time tagging uses sonic cues as proxies for the temporal distribution of sonic events. The next section provides multiple examples of experiments that employ time tagging and the stripe metaphor for mixing sonic resources.

## 3. Time tagging experiments

Two generations of prototypes were designed and deployed [5, 8]. As an initial validation process, Keller et al. (2009) used an emulation of a first-generation mixDroid prototype (mixDroid 1G) for the creation of a complete musical work [1]. The procedure encompassed several mixing sessions. The mixDroid 1G prototype was used in the emulation mode on a laptop computer and was activated through pointing and clicking with an optical mouse. Several dozens of sound samples were used, with durations ranging from less than a second to approximately two minutes. The temporal structure of the mix was based on the temporal characteristics of the sonic materials (biophonic sounds). The result was a seven-minute stereo sound work – *Green Canopy On The Road* – the first documented ubiquitous music work, premiered at the twelfth Brazilian Symposium on Computer Music, held in Recife, PE [1].

Focusing on the demands of naive participants in everyday contexts, a second study [9] comprised creative activities in public settings – at a shopping mall, at a busy street and in a quiet area featuring biophonic sounds – and in private settings – at the home of each participant and at a studio facility. Six subjects participated in 47 mixing sessions using samples collected at two outdoors sites comprising urban sounds and biophonic sources. Creativity support was evaluated by means of a creative-experience protocol encompassing six factors: productivity, expressiveness, explorability, enjoyment, concentration, and collaboration (CSI-NAP v. 0.1 – [10]). Outdoor sessions yielded higher scores in productivity, explorability, concentration and collaboration when compared to studio sessions. Compound effects of sound sample type and activity location were observed in the explorability factor when biophonic sound samples were used. Similar effects were detected on explorability, productivity and concentration in the conditions employing urban sounds.

A third study [11] made use of recorded vocal samples created by the participants. In order to untangle the effects of place and activity type, three conditions were studied: place, including

domestic and commercial settings; activity type, i.e. imitative mixes and original creations; and body posture, realizing the mix while standing or sitting. Ten subjects took part in an experiment encompassing 40 interaction sessions using mixDroid. Subjects created mixes and assessed their experiences through a modified version of the CSI protocol applied in the previous studies [10]. Explorability and collaboration factors yielded superior scores when the activities were carried out in domestic settings.

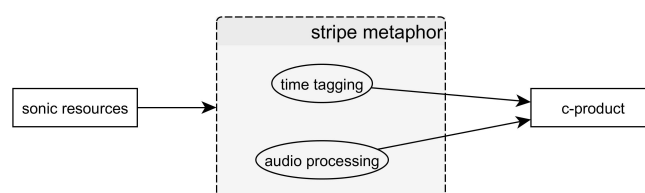
The results highlighted the impact of the venue on the support of everyday creative experiences. The outdoor spaces were preferred by the participants of the second study and domestic settings got slightly higher ratings in the third study. While the profile of the subjects impacted the outcome of the third study, this trend was not confirmed by the second study's results. Hence, the main conclusion to be drawn from these studies points to the impact of the venue on the subjects' evaluation of the creative experience. Both their ability to explore the potential of the support metaphor and their ability to collaborate were boosted by domestic settings and by outdoor settings.

#### 4. Fostering professional creativity in everyday settings, the stripe

Despite the positive outcomes of the experiments involving time tagging support for novices, no attempt was made to address the needs of professional participants. Given the different requirements of musicians and non-musicians [12], it would not be surprising to find that effective support for novices does not meet the expectations of professional usage. In this section, I describe a new metaphor based on time tagging and a methodological strategy that incorporates an ecology of devices to support creative activities by both musicians and laypeople in everyday settings.

The second generation of mixDroid prototypes features a new interaction mechanism: the stripe (figure 1). The stripe acts as a functional unit that features both interaction support and audio manipulation. This metaphor ties to the

sonic sample the functionality previously linked to the audio channel in analogue systems. The objective is to allow for synchronous interaction with a large number of elements to overcome the screen-size limitations of small devices. Stripes enable mixing using both hands. The amount of active stripes depends on the device's computing power and on the participant's cognitive abilities. Thus, similarly to previous time-tagging implementations [8], devices with low computational resources can be used for complex creative activities in everyday contexts.



**Figure 1: The stripe metaphor: bringing together audio processing and creative decision making on a single functional unit.**

The *stripe* acts as an entry point to the sound data. Each stripe displays basic information on the sample being handled, including the file name, the total running time, the current time and the execution state [5] (see figure 5). Each sound file linked to a stripe is processed independently. By linking the interaction mechanism with the sound sample, the stripe releases the user from the requirement of dealing with multiple samples as a block (as it is the case in the mixing-console metaphor that has the audio channel as its basic functional unit). Synchronous mixing of multiple sound sources is supported without compromising the parametric independence of each source. From the perspective of the user, sounds that demand fast interaction can be placed on stripes that are close to each other. This flexibility, combined with the ability to select stripes through scrolling, should grant quick access to a large number sonic items.

#### 5. Provocative Synthesis II: a stripe metaphor study

The study *Provocative Synthesis II* addressed the impact of the stripe metaphor in musical activities involving both musicians and novices.

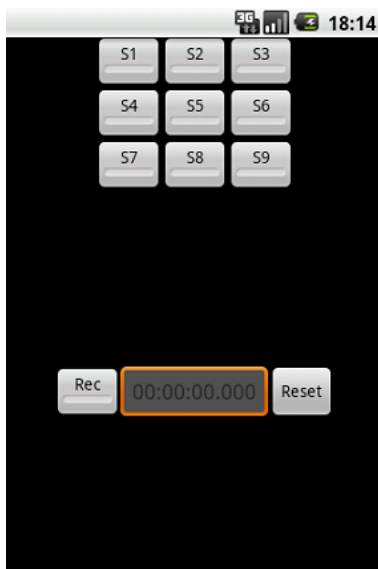


Figure 2: mixDroid 1G, on emulator.

The study encompassed musicians performing amplified electric instruments and casual users triggering audio sequences on mixDroid 2G CS. This proposal forms part of a series of ubimus experiments devised by the musician Edemilson Ferreira [11, 13].

*Settings and equipment.* The study was carried out at an audio and musical equipment store located in downtown Rio Branco, AC, Brazil. The hardware included a portable computer, a six-channel JamHub mixer, and stereo headphones for each of the participants. The JamHub system [14] features independent returns through headphones for each user. The output from the mixer was routed to the computer and sound levels were monitored by the researcher (Figure 4). The prototype mixDroid 2G CS was used on an Iconia One Acer tablet, running the Android 4.1 operating system. The musicians played electric guitar and electric bass.

*Subjects profile.* Twelve subjects participated in the sessions, including 6 musicians and 6 laypeople (table 1). Their average age was 28.5 years with a standard deviation of 6.76 years. Three of the six self-described musicians had no formal study but reported ten or more years of proficiency playing either electric bass or electric guitar.

N	age	men	women	musicians	non-musicians	formal study
12	28.5 ± 6.76	7	5	6	6	3 years

*Procedures.* The eleven improvisational ses-

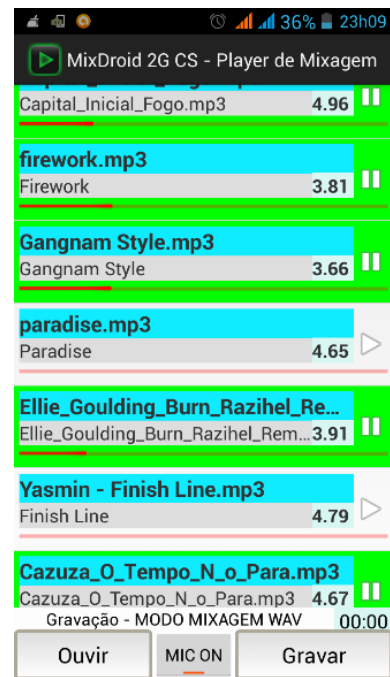


Figure 3: mixDroid 2G, the stripe metaphor.

sions had an average of 4 participants per session. Each activity lasted approximately two minutes. Lay participants sat on benches, while the musicians preferred to play standing. Musicians were free to improvise within the bounds of the rhythmic reference laid out by the soundtrack and by the verbal instructions. Throughout the sessions, the mixDroid player was responsible to start the sonic exchanges. All sonic results were recorded as PCM uncompressed audio files.

As in previous creativity assessment experiments [5, 12, 9], we employed the CSI-NAP to collect data on the creative activity and the creative products. Two factors assess the creative products (relevance and originality). The other four factors target the experience involving the settings, the tools and the participants' experience (easiness of use – the inverse of cognitive effort; focus on activity; fun or enjoyment during the activity; productivity – whether the activity and the result were considered productive; and collaboration – involving the support for social interaction among the participants). The 5-point Likert scale adopted varies from 'I strongly disagree' (-2) to 'I strongly agree' (+2). Zero stands for no preference or no answer. All the participants filled the forms immediately after each session.

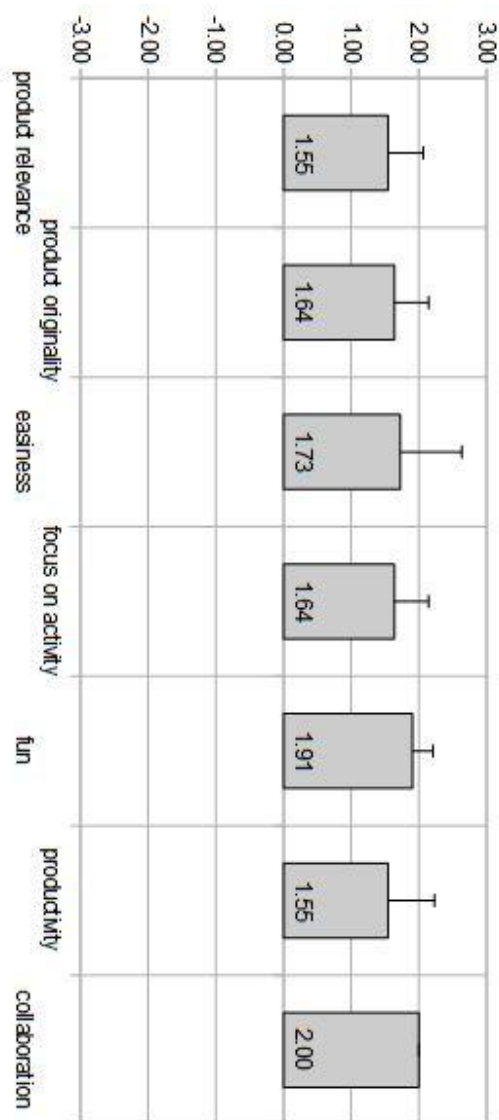




**Figure 4: Subjects participating in *Provocative Synthesis II* at an audio and music equipment store.**

*Results.* The overall results indicate a positive assessment of the experience (Figure 6). Strikingly, both musicians and non-musicians gave the highest rating to collaboration support ( $2.00 \pm 0.00$ ). Enjoyment was also highly rated with little variation among sessions ( $1.91 \pm 0.30$ ). Easiness of use got high ratings by all subjects except one ( $1.73 \pm 0.90$ ). The other two descriptors of the experience got positive ratings but higher standard deviations: focus ( $1.64 \pm 0.50$ ) and productivity ( $1.55 \pm 0.69$ ). Finally, the musical product was described as being creative, but this result was not uniform across subjects: relevance ( $1.55 \pm 0.52$ ) and originality ( $1.64 \pm 0.50$ ).

A description of the observations done throughout the sessions may help to give context to the results. The choice of the experimental settings somehow facilitated the initial contact with the subjects. In contrast with previous experiments in everyday musical creativity [11, 15, 16, 9], *Provocative Synthesis II* involved interactions among professional musicians and laypeople. Previous experiments focusing on creative activities outside of musical venues involved difficulties in drafting participants for the tasks.



**Figure 5: Results of the *Provocative Synthesis II* experiment for 11 iterations by 4 subjects per session. Data collected through the CSI-NAP.**

This was not the case in this experiment. During the activities, we observed that a majority of store visitors were curious and interested in participating in the experience. While novices are generally reluctant to get involved in activities with non-musicians, in this case the possibility of participating in a creative activity with experienced musicians may have sparked their interest.

*Limitations.* Despite the promising avenue opened by the initial results, several difficulties still need to be addressed. Given the amount of equipment involved, technical preparations become tricky. The multi-user mixer is an effec-

tive bridge to integrate instrumental sources with ubiquitous music tools, such as mixDroid. But the dependence on wires and plugs compromises the portability of the system. A combination of wireless connectivity and software-based multi-user mixing might help to increase the mobility of the setup. Another limitation of the study is the small number of sonic resources used by the casual participants. This variable may be related to the profile of the user or to the time available for exploration. In this case, the objective was to assess the ability of the participants to create collaborative sonic products without extended preparation. Hence, it is not possible to determine whether the subjects needed more time or a different kind of support.

*Contributions.* The high ratings given to the collaboration factor indicate that both the musicians and the untrained subjects felt they were contributing effectively to the musical result. Neither the characteristics of the settings nor the creative products conform to the expected patterns of a professional musical experience. Consequently, these results cannot be classified as a professional-creativity musical outcome (cf. [17] for a theoretical discussion of this issue). Furthermore, the high level of collaboration reported here was not observed in other everyday creative activities [11, 9]. The results may be interpreted as an indication that the musical experiences at the fringe of professional and everyday creativity open opportunities for effective contributions both from musicians and novices.

## 6. Challenges for future ubimus endeavors

A recent review of musical creative practices [18] mentions four trends that demand stronger theoretical and methodological frameworks: (1) change of focus from creative products to processes [19]; (2) increased reliance on information technology support [20]; (3) increased importance of local resources in creative activities [21]; (4) a shift from prescriptive models to descriptive and predictive models [22, 23, 18]. The design strategies discussed in this paper provide examples of creative decision making grounded on local material resources. The two support

metaphors proposed in this paper enhance the available set of techniques to handle these resources.

A good example of the importance of creative processes is provided by the lay-musician interaction phenomenon observed in the stripe metaphor study. Despite the increased levels of engagement by naive participants, it is not clear how to address the demands of professional stakeholders to achieve creative outcomes. The strategy employed in this study involved restrictions on the type of musical material – through verbal instructions provided at the outset – and freedom of action for casual participants. Effective support may involve a combination of fixed resources and open procedures. How much flexibility and what level of guidance are necessary are questions to be answered through multiple iterations of designs and experiments.

The emergence of everyday musical creativity as a social phenomenon worth of study provides grounding for Truax's assertion that current musical practices have raised the demands for technological support. Little-c music can hardly be conceptualized without the existence of mobile and ubiquitous technology. Wireless networks, portable devices and embedded technologies constitute the venues that foster creative music making almost anywhere [24]. The time tagging studies reviewed in this paper indicate the need of a broader understanding of the support requirements for everyday creativity. A variety of environmental impacts on the creative activities were consistently documented, as enablers for creative action and as negative influences on the participants performance. While outdoor spaces got positive evaluations when compared to studio settings and domestic spaces were chosen over commercial spaces, the assessments were not uniform across all factors.

An important flag raised by the *Provocative Synthesis II* study is the potential exploration of an inclusive form of social engagement involving both proficient partners and casual participants in ubiquitous musical activities. Acoustic musical instruments demand long periods of training to achieve minimally rewarding musical results. This practice is aligned with the needs of pro-

fessional creative results. The concept of everyday musical creativity that emerges from recent ubimus studies widens the geographical availability of spaces for music making by including creative phenomena which are not linked to artistic venues and enhances the access by stakeholders that have been traditionally excluded from creative practice. The results of the study point to good levels of engagement when both musicians and casual participants are involved. But support for lay-musician interaction may imply tailoring for specific needs.

## References

- [1] Damián Keller, Ariadna Capasso, Patricia Tinajero, Luciano Vargas Flores, and Marcelo Soares Pimenta. Green canopy: On the road [ubiquitous music work]. In *Proceedings of XII Brazilian Symposium on Computer Music (SBCM 2009)*. Porto Alegre, RS: SBC, 2009.
- [2] Damián Keller, Ana Elisa Bonifácio Barros, Flávio Miranda Farias, Rafael Vasconcelos Nascimento, Marcelo Soares Pimenta, Luciano Vargas Flores, Evandro Manara Miletto, Eduardo Aquiles Affonso Radanovitsck, Rafael Oliveira Serafini, and José F. Barraza. Ubiquitous music: Concept and background. In *Proceedings of the National Association of Music Research and Post-Graduation Congress - ANPPOM*, pages 539–542. National Association of Music Research and Post-Graduation (ANPPOM), Goiânia, GO: ANPPOM, 2009.
- [3] M. S. Pimenta, L. V. Flores, A. Capasso, P. Tinajero, and D. Keller. Ubiquitous music: concept and metaphors. In R. R. A. Farias, M. Queiroz, and D. Keller, editors, *Proceedings of the Brazilian Symposium on Computer Music (SBCM 2009)*, pages 139–150. Recife, PE: SBC, 2009.
- [4] Damián Keller, Daniel Luis Barreiro, Marcelo Queiroz, and Marcelo Soares Pimenta. Anchoring in ubiquitous musical activities. In *Proceedings of the International Computer Music Conference*, pages 319–326. Ann Arbor, MI: MPublishing, University of Michigan Library, 2010.
- [5] Flavio Miranda Farias, Damián Keller, Floriano Pinheiro Da Silva, Marcelo Soares Pimenta, Victor Lazzarini, Maria Helena Lima, Leandro Costalonga, and Marcelo Johann. Everyday musical creativity support: mixdroid second generation. In Damián Keller, Maria Helena Lima, and Flávio Schiavoni, editors, *Proceedings of the V Workshop on Ubiquitous Music (V UbiMus)*. Vitória, ES: Ubiquitous Music Group, 2014.
- [6] Marcelo S. Pimenta, Evandro M. Miletto, Damián Keller, and Luciano V. Flores. *Technological support for online communities focusing on music creation: Adopting collaboration, flexibility and multiculturalism from Brazilian creativity styles*, volume Cases on Web 2.0 in Developing Countries: Studies on Implementation, Application and Use, chapter 11. Vancouver, BC: IGI Global Press, 2012.
- [7] Damián Keller, Nuno Otero, Victor Lazzarini, Marcelo Soares Pimenta, Maria Helena Lima, Marcelo Johann, and Leandro L. Costalonga. *Interaction aesthetics and ubiquitous music*, volume Creativity in the Digital Age of *Series on Cultural Computing*, pages 91–105. Berlin and Heidelberg: Springer, 2015.
- [8] Eduardo Aquiles Affonso Radanovitsck, Damián Keller, Luciano Vargas Flores, Marcelo Soares Pimenta, and Marcelo Queiroz. mixdroid: Time tagging for creative activities. In L. Costalonga, M. S. Pimenta, M. Queiroz, J. Manzolli, M. Gimenes, D. Keller, and R. R. Farias, editors, *Proceedings of the XIII Brazilian Symposium on Computer Music (SBCM 2011)*. Vitória, ES: SBC, 2011.
- [9] Floriano Pinheiro da Silva, Damián Keller, Edemilson Ferreira da Silva, Marcelo Soares Pimenta, and Victor Lazzarini. Everyday musical creativity: Exploratory study of ubiquitous musical activities. *Música Hodie*, 13:64–79, 2013.
- [10] D. Keller, F. Pinheiro da Silva, B. Giorni, M. S. Pimenta, and M. Queiroz. Spatial tagging: an exploratory study. In L. Costalonga, M. S. Pimenta, M. Queiroz, J. Man-

- zulli, M. Gimenes, D. Keller, and R. R. Farias, editors, *Proceedings of the 13th Brazilian Symposium on Computer Music (SBCM 2011)*. Vitória, ES: SBC, 2011.
- [11] Damián Keller, Floriano Pinheiro da Silva, Edemilson Ferreira da Silva, Victor Lazzarini, and Marcelo Soares Pimenta. Opportunistic design of ubiquitous music systems: The impact of anchoring on creativity. In Edilson Ferneda, Giordano Cabral, and Damián Keller, editors, *Proceedings of the XIV Brazilian Symposium on Computer Music (SBCM 2013)*. Brasília, DF: SBC, 2013.
- [12] Maria Helena de Lima, Damian Keller, Marcelo Soares Pimenta, Victor Lazzarini, and Evandro Manara Miletto. Creativity-centred design for ubiquitous musical activities: Two case studies. *Journal of Music, Technology and Education*, 5(2):195–222, 2012.
- [13] Edemilson Ferreira, Damián Keller, and Maria Helena Lima. Esboços sonoros em música ubíqua: Perspectivas educacionais. *Sonic Ideas*, 2015. Morelia, México: CM-MAS.
- [14] Paul White. Jamhub: Personal monitor system, 2010.
- [15] Damián Keller, Flávio Miranda Farias, Edemilson Ferreira da Silva, Floriano Pinheiro da Silva, Marcelo Soares Pimenta, Victor Lazzarini, M. H Lima, Leandro Costalonga, and Marcelo Johann. Perspectives in ecological cognition for ubiquitous music: everyday creativity support challenges. In Damián Keller, Maria Helena de Lima, and José Fornari, editors, *Challenges in ubiquitous music research*, volume Anais do XXIV Congresso da Associação Nacional de Pesquisa e Pós-Graduação em Música (Proceedings of the XXIV National Association of Research and Graduate Studies in Music) (ANPPOM 2014). São Paulo, SP: ANPPOM, 2014.
- [16] Damián Keller and Maria Helena de Lima. *Supporting everyday creativity in ubiquitous music making*, volume Trends in Music Information Seeking, Behavior, and Retrieval for Creativity. Vancouver, BC: IGI Global Press, 2016.
- [17] Damián Keller, Victor Lazzarini, and Marcelo Soares Pimenta. *Ubiquitous Music*, volume XXVIII of *Computation Music Series*. Berlin and Heidelberg: Springer International Publishing, 2014.
- [18] Damián Keller, Victor Lazzarini, and Marcelo S. Pimenta. Ubimus through the lens of creativity theories. In Damián Keller, Victor Lazzarini, and Marcelo S. Pimenta, editors, *Ubiquitous Music*, Computational Music Science, pages 3–23. Berlin and Heidelberg: Springer International Publishing, 2014.
- [19] Alan Marsden. "what was the question?": Music analysis and the computer. In L. Gibson and T. Crawford, editors, *Modern Methods for Musicology: Prospects, Proposals, and Realities*, Digital Research in the Arts and Humanities, chapter 10, pages 137–153. London: Ashgate Publishing, 2012.
- [20] Barry Truax. Genres and techniques of soundscape composition as developed at simon fraser university. *Organised Sound*, 7(1):5–14, 2002.
- [21] Damián Keller. *Sonic Ecologies*, volume Sound Musicianship: Understanding the Crafts of Music, pages 213–227. Newcastle upon Tyne, UK: Cambridge Scholars Publishing, 2012.
- [22] Daniel Luís Barreiro and Damián Keller. *Composing with sonic models: fundamentals and electroacoustic applications*, volume Criação Musical e Tecnologias: Teoria e Prática Interdisciplinar of *Pesquisa em Música no Brasil*, pages 97–126. Goiânia, GO: Editora ANPPOM, 2010.
- [23] S. Ferraz and D. Keller. Preliminary proposal of the mdf model of collective creation. *Cadernos de Informática*, 8(2):57–67, 2014.
- [24] Marcelo S. Pimenta, Damián Keller, Luciano V. Flores, Maria Helena Lima, and Victor Lazzarini. *Methods in Creativity-Centred Design for Ubiquitous Musical Activities*, pages 25–48. Springer International Publishing, 2014.

# Design and implementation of an open-source subtractive synthesizer on the Arduino Due platform

Rodolfo Pedó Pirotti<sup>1</sup>, Marcelo Johann<sup>1</sup>, Marcelo Pimenta<sup>1</sup>

<sup>1</sup>Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)  
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

rodolfo.p.pirotti@gmail.com, {johann, mpimenta}@inf.ufrgs.br

## ABSTRACT

In this paper we present the design of a digital subtractive synthesizer using fixed-point arithmetic on the Arduino Due platform. Our main contribution is to show that a fully functional instrument of this type can run on a cheap and widely accessible processor. We have implemented oscillators with anti-aliasing algorithms, resonant filters, an envelope generator, a delay effect, a MIDI interface and a keybed scanner, therefore making a complete playable instrument. The implementation uses object orientation to create software modules replicating those classic analog synthesizer functions. With this approach, we have a modular software system that can be easily extended and adapted for new functionalities. An external DAC was used to provide the high-quality audio output of 16 bits at 48KHz. In addition to this, we also included an additive synthesis organ, demonstrating the possibility of having two important synthesis methods at the same time on the Due board. With this open and public design, we intend to contribute to the maker movement and encourage new and innovative implementations in this area.

## 1. Introduction

A significant number of audio synthesis techniques are used nowadays on professional musical instruments and home-made projects as well. Among these, the most fundamental are the subtractive synthesis and the additive synthesis. Subtractive synthesis became very popular with the analog synthesizers of the 60's and 70's, which used hardware modules that could be connected together, with a modular concept. The introduction of this type of electronic instrument brought new areas of possibility to sound creation and musical performance – the adoption and

popularity of analog synthesizers grew up fast with the synthesizers launched on those years.

Nowadays, on the digital domain, virtual analog instruments, which are digital and software emulation of analog synthesizer functions, are common, either as separate instruments or as part of most digital synthesizers. Nevertheless, the prices of these instruments are often high.

The complexity involved in the design of professional digital musical instruments is naturally increasing as the technology progresses. It usually involves the design of ASICs (application-specific integrated circuits), or programming of complex DSPs (Digital Signal Processors), lots of memory and all the sound capture and processing tasks required to prepare samples, design of complex PCBs (Printed Circuit Boards) and so on. In some sense, the complexity usually employed makes us believe that this is absolutely necessary to get a functional instrument.

On the other hand, smaller processors have also evolved tremendously, and nowadays, for a small budget, there is available a large number of open-source platforms. Boards like Arduino, Raspberry Pi and Beagle Bone provide computing, processing power and interfaces on a ready-to-go board that can be used even by people with almost no knowledge of engineering and electronic components, and the number of users of these platforms is growing every year [1]. Part of this increasing number of projects and boards is related to the DIY culture and maker movement, emerging movements in which people aim to create, design, build or modify products, equipment, home items, making art projects, music and several other things.

In the audio realm, there have been many attempts to implement basic audio synthesis us-

ing platforms such as the Arduino. But until recently, it was not possible to implement full-featured instruments with an audio quality considered as "professional", here defined as being able to generate and output audio with at least 16-bit resolution, 48 kHz sample rate and no output aliasing, because the basic boards that were widely available lacked resources and computing power for that. In other words, the audio synthesis implementations for the Arduino Uno, for example, are very interesting from the point of view of learning and experimentation, but too limited to be used to implement a "real instrument", something with an audio quality that would be used by a trained musician for traditional performances or studio recording. They present too little resolution, limited bandwidth, lots of noise and aliasing and so on.

In this work, we wanted to investigate the hypothesis that it is possible to design and implement a fully functional subtractive synthesizer with good quality audio output using the still limited processing resources of the Arduino Due platform, in opposition to the premise that a high-quality music synthesizer needs dedicated and expensive processors and electronic components. By achieving this goal, this design could become a reference to be used by others to build their own synthesizer on a low-budget hardware.

The rest of this paper is organized as follows: Section 2 presents a comparison with related work and the specific goals of this paper. Section 3 presents the design of subtractive synthesizer modules, an overview of software modules integration and considerations about the additive synthesizer. Section 4 presents analysis and measurements made with the implementation. Finally, in Section 5 we describe some conclusions and discussions about future ideas.

### 1.1 The Arduino Due Platform

Arduino is an open-source prototyping platform based on flexible and easy-to-use hardware and software. Its usage has been growing on recent years by students, professionals and amateurs in many areas. Comparing to other open-source platforms listed before (Raspberry PI and Beagle Bone), Arduino is more suitable for low-level applications, and its learning curve is usually faster comparing to others [1].

The Arduino Due has a 32-bit ARM micro-controller running at 84 MHz and 96 kB of RAM memory. With these specifications, unlike simpler boards as Arduino Uno or Nano, we estimated that serious audio applications could run output audio with minimum professional audio quality (16 bits / 48 kHz).

## 2. Context

In order to provide the context to our contribution, we have listed a few previous related works with implementation of subtractive synthesizers and classified them according to the following criteria:

- Build complexity: how hard it is for someone with small knowledge of electronic and computing to build;
- Total cost: total cost to build and use the synthesizer;
- Usage purpose: if for learning purpose only, with no commitment regarding the output quality; if for usage as a real music instrument;
- Output quality: if the output fits the minimum requirements of 16 bits of audio resolution, sample rate above Nyquist frequency (sample rate at least twice the highest harmonic – around 17 kHz for additive synthesis organ and 44.1 kHz for virtual analog) and no output aliasing.

Table 1 shows a comparison of related works. By looking at their characteristics, we can observe that our proposal differs from others in many aspects. The work of [2] presents a subtractive synthesizer implementation for accessible platforms, but with low audio quality output and no anti-aliasing algorithms. In [3], authors only propose how to design an instrument, without showing the actual implementation, whereas in [4] it is necessary to pay for the project. There are other projects like [5], which presents a design suitable for a PC computer, and [6,7], as examples of complete analog synthesizer designs. In such cases, they are either too expensive and with a high complexity to build, or require a full personal computer to run, different from the purpose of this work.

Author	[2]	[3]	[4]	This work
Build complexity	Medium	NA	Easy	Medium
Total cost	Cost of Arduino Uno (aprox. US\$ 22), plus few external components	NA	Product sold by US\$ 266	Cost of Arduino Due (aprox. US\$ 30), plus few external components
Usage purpose	Learning	NA	Learning	Instrument
Output quality	Poor	NA	Poor	High

Table 1. Related work comparison

## 2.1 Specific goals

After the comparison, we define the specific goals of this work:

- To design and develop a subtractive synthesizer using classic modular analog synthesizers as reference, on a cheap, widely available and easy to use platform – Arduino Due;
- To identify and use efficient algorithms that fit in the limited processing capacity of the chosen platform.
- To have a modular software design that makes it easy to change and add features as needed;
- To add a good-quality audio output, so the synthesizer can be used as a real musical instrument;
- To include on the design an additive synthesis, based on [8], to show that it is possible to have two fundamental synthesis methods programmed at the same time on a small budget platform;
- To contribute to the maker movement by sharing the design, encouraging new and innovative implementations in this area and increasing the usage and access to (electronic) musical instruments and the musical creativity.

## 3. Synthesizer Design

We started the design by establishing some guidelines so that the implementation could be successful, easily tested, and further extended. We defined that:

- for mathematic calculation, we must use fixed point arithmetic instead of floating point arithmetic, as the Arduino Due processor does not have floating point unit.
- for subtractive synthesis, the software model should use an object-oriented design, making

separate classes which mimic the functions of analog synthesizers (their modules);

- for test and simulation, to implement mock or hardware abstraction functions that enable the test and simulation of algorithms on a standard PC;

### 3.1 Time-critical, time-accurate and house-keeping tasks

This organization was already proposed by [8], and is used in our implementation to efficiently compute the necessary functions according to their criticality. The time-critical tasks include the code for the audio processing, like waveform generation, filter and effect calculation. This is the code that needs to run at the exact sample rate to produce each audio sample.

Time-accurate tasks include tasks whose updates are much less frequent, but depend on correct timing. They do not need to be computed at each sample, but still need to be accurately evaluated over time. They typically include the envelope generator and other low-frequency modulation functions. Part of this code might still be computed at the main loop, but at least the time-accurate interruption needs to keep track of the time elapsed by means of one or more counters, as needed.

It is important to note that the time-accurate tasks must be implemented as an interrupt with a lower priority than the time-critical code, otherwise they will interfere with it.

Finally, housekeeping tasks include the keybed scanning, switches and analog input readings, as well as any general control code. This last set of tasks do not need a time-accurate execution and can be slightly delayed as long as we keep their periodicity under acceptable values so not to impact the playing experience in a bad way.



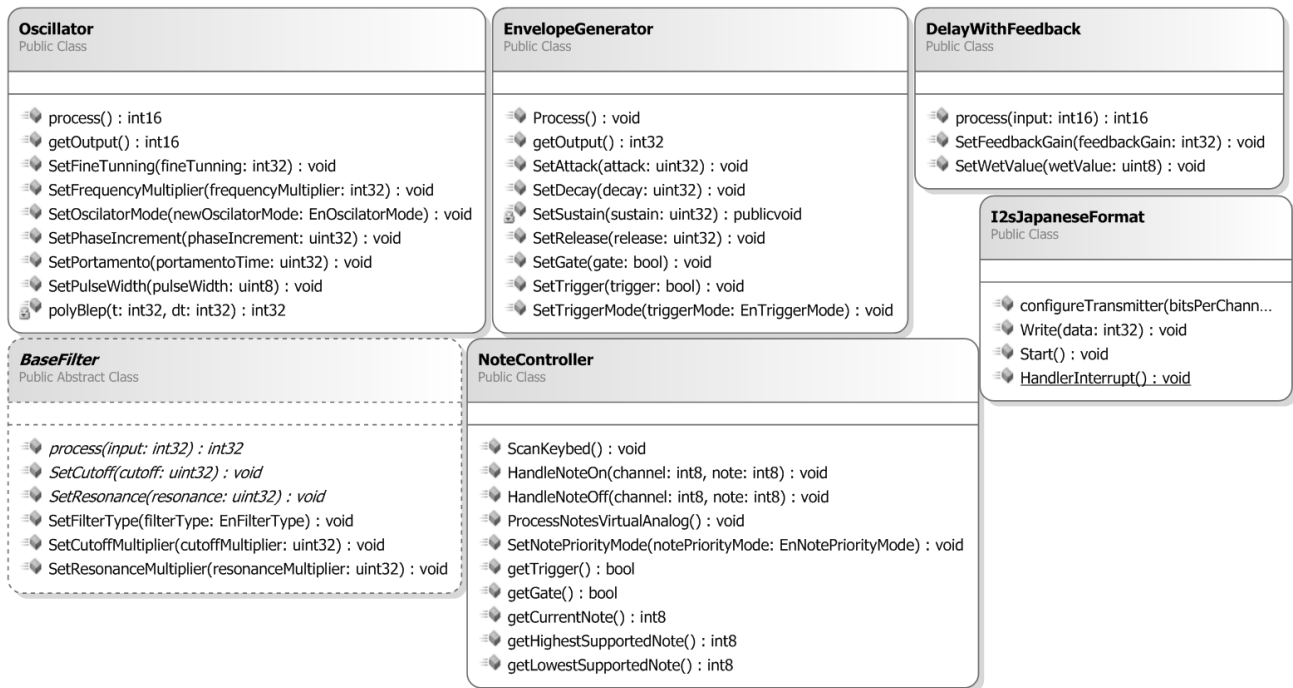


Figure 1. Main classes developed for the subtractive synthesizer

With this structure in mind, the code is composed of the classes shown in Figure 1, which will be described in the next sections.

### 3.2 Oscillator module

According to [9] and [10], the oscillator module converts a voltage value to a waveform output on a specific frequency. Usually it also has control inputs such as to modulate the main frequency and select waveform type.

In our implementation, we've created a software class that implements the oscillator, with features such as waveform type selection, portamento (a feature usually associated to the keyed on classic analog synthesizers, but we chose to move it to the oscillator class), frequency modulation (to use together with an LFO), fine tuning and octave selection.

One of the key points of the oscillator class is the waveform generation algorithm. The Numeric Controlled Oscillator (NCO) [11] is the simplest approach and does not require a lot of calculation, but it produces undesired aliasing in the output signal for pulse and sawtooth waves. A comparison of anti-aliasing algorithms is presented on [12], and based on that we chose to use the Polynomial Bandlimited Step Function (PolyBLEP) algorithm for pulse and sawtooth waves, while the sine wave uses the wavetable method [13], as there is no aliasing on this

method for sine waves. The PolyBLEP algorithm was implemented using fixed-point arithmetic and 32-bits integer types.

### 3.3 Envelope Generator module

According to [9] and [10], the Envelope Generator (EG) module generates a signal used to control other modules in order to give a contour to some parameter, like the signal amplitude or filter cutoff frequency.

Our design has control inputs to configure *Attack*, *Decay*, *Sustain* and *Release* values, plus *Gate* and *Trigger* signal simulation. A seventh input is used to configure the EG behavior (*SetTriggerMode*). The output value is a 32 bits fixed point value, with 16 bits of fractional part. The output value is always less or equal to 65536 (equivalent to 1.0), and can be used as a multiplier to another module parameter. Figure 2 shows the state machine of this module.

### 3.4 Filter module

For the filter module, we created a base class to be an interface class to allow different filter implementations. On our current prototype, there are two different low pass filters, one based on [5] (a fourth-order filter) and other based on [14] (a fourth-order filter with resonance based on Robert Moog's filter used on Moog synthesizers).



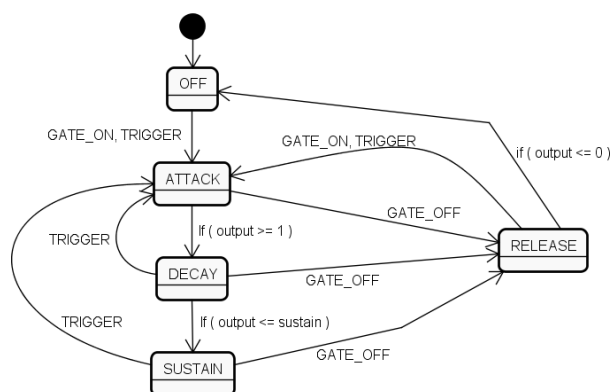


Figure 2: Envelope generator state machine

### 3.5 Delay effect and note controller modules

The delay effect module is a simple delay with feedback effect, used to create a repeating and decaying echo in the sound. The note controller module was created to be responsible for managing the notes being played and generate the gate and trigger signals. It has methods to process MIDI events (note on and note off) and a method to scan a keyboard.

### 3.6 External DAC

Arduino Due processor has a 12 bits integrated DAC (digital to analog converter), which is good for general applications and even some experimental audio applications, but in order to improve the audio processing and reduce output noise, we chose to use an external 16 bits DAC.

The DAC used is the TDA1543A, a dual-channel 16 bits DAC for usage in hi-fi applications. Using the code provided by [15] as reference, we implemented our own class responsible to configure and send data to the DAC.

### 3.7 Module integration

Each software module was created as a separate class and can be instantiated as different objects. This approach makes it possible and easy to add (or remove) oscillators, envelope generators, filters and effects (and other features) to the synthesizer, keeping in mind the processing time of each module.

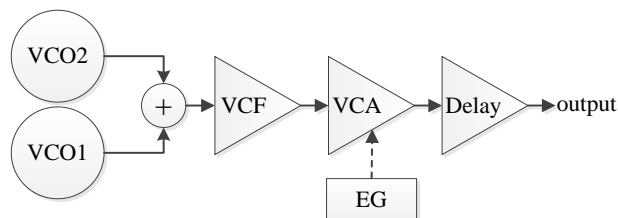


Figure 3: Typical synthesizer diagram

Figure 3 shows an example of a typical connection between analog synthesizer modules. The code below is the equivalent code in our design to implement the same architecture and connections.

```
output = pVCO1->process() + pVCO2->process();
output = pFilter->process( output );
output = output * pVcaEg->getOutput() >> FRACT_WIDTH;
output = pDelay->process( output );
```

### 3.8 Testing and simulation

This implementation runs on an embedded system with limited resources and interfaces. This would make it harder to debug and validate the algorithms. To improve testing and simulation capabilities, we created empty code definitions and calls to hardware related interfaces. This enables us to build the code using a standard compiler like GCC or MS Visual Studio on a standard PC, taking advantage of all debugging tools.

### 3.9 Additive synthesis organ

When we started the development of the subtractive synthesizer, we realized that the most critical concern is the processing speed, mainly because of the lack of a hardware floating point unit. After the first code implementations and tests, we observed that the amount of flash and RAM memory available on the Arduino Due processor was much more than needed for our proposal. We decided therefore to implement an additive synthesis organ, based on the implementation described in [8], and integrate it to the subtractive synthesizer.

The implementation, based on classic Hammond organ design, includes the functions of 9 drawbars, 61 notes with full polyphony, 96 oscillators (compared to 91 implemented on Hammond organs), vibrato and tremolo effects. For this implementation, the output sample rate

was reduced to 24 kHz, but it is enough for the organ design because its highest oscillator runs at 8372 Hz.

The integration of subtractive and additive synthesis cannot be simultaneously played, as each audio generation tasks takes up almost all the processing power. On the other hand, they can coexist programmed at the Arduino Due board at the same time, and they can be switched on the fly, e.g., the user can select one or the other mode and start playing right away by changing a simple switch.

### 3.10 Block diagram

Figure 4 shows a block diagram of the final synthesizer. The Arduino Due board is the main component. In addition to this, there's an array of potentiometers, used to control analog parameters in realtime, an array of digital switches, used to enable and disable features, the digital-to-analog converter, the MIDI input circuit, and digital pins connected to the keybed, some configured as output and others configured as inputs (the keybed works as a matrix with 8 columns and 7 rows).

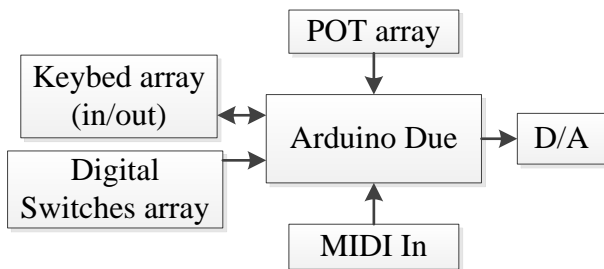


Figure 4: Hardware block diagram

Figure 5 shows one potentiometer input. The POT array shown on Figure 5 is an array of 11 potentiometer inputs, each one connected on a different Arduino analog input.

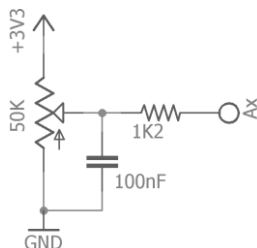


Figure 5: Potentiometer input

The digital switches array is a group of 16 digital switches, connected to Arduino digital inputs. Figure 6 shows an example of 1 input.

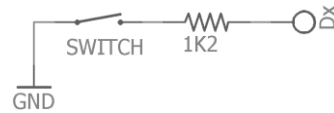


Figure 6: Digital switch input

The D/A is the circuit used for the TDA1543A DAC component, shown on Figure 7.

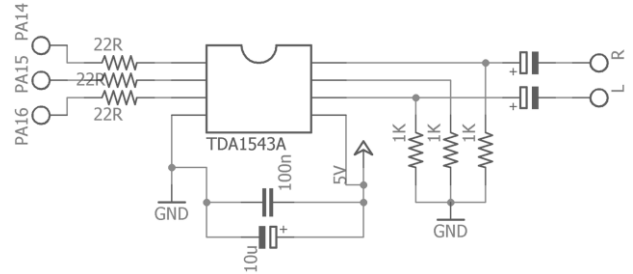


Figure 7: TDA1543A digital-to-analog converter

Figure 8 shows the MIDI input circuit.

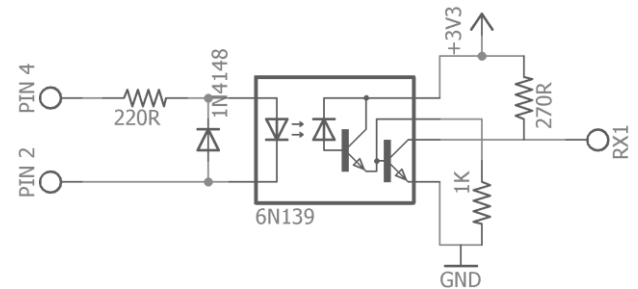


Figure 8: MIDI In circuit

For the keybed scanning, there are 7 digital ports configured as output and 8 digital ports configured as input. Figure 9 shows one of the output digital ports (Dx is Arduino output and Ko is keybed), and Figure 10 shows one of the input digital ports (Dx is Arduino input and Ki is keybed).

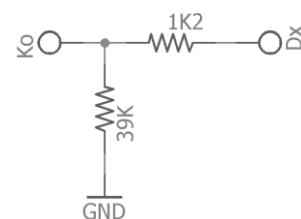


Figure 9: Arduino output for keybed



Figure 10: Arduino input for keybed

## 4. Features and output analysis

We present here some analysis regarding important areas of the implementation.

## 4.1 Output aliasing check

The anti-aliasing algorithm was chosen based on the analysis of [12]. The PolyBLEP algorithm seemed to be a nice approach, with low processing overhead and good output results. Figure 11 shows the frequency spectrum of the output **without** the PolyBLEP algorithm for a square wave, C note, frequency of 2093 Hz.

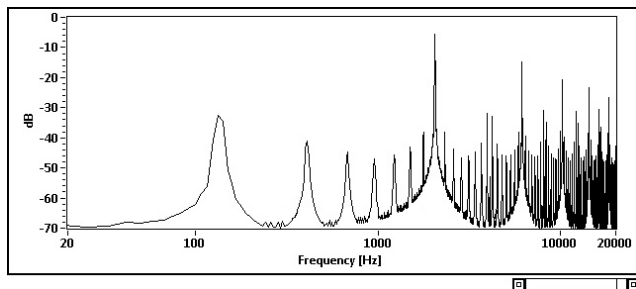


Figure 11: Frequency spectrum without PolyBLEP

Figure 12 shows the frequency spectrum of the output with the PolyBLEP algorithm, for a square wave, C note, frequency of 2093 Hz. It is clearly visible the reduced aliasing caused by the PolyBLEP addition.

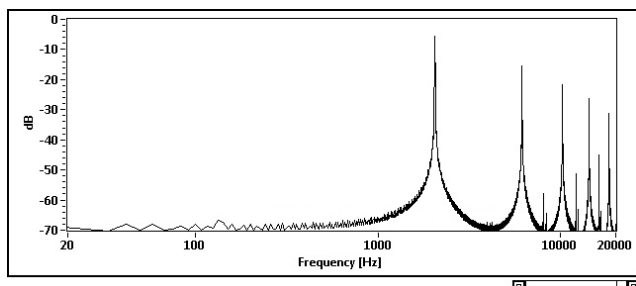


Figure 12: Frequency spectrum with PolyBLEP

## 4.2 Filter spectrogram

Figure 13 shows the spectrogram of one of the filters implemented [14] for a single note using square wave, while increasing the cutoff frequency and the resonance control.

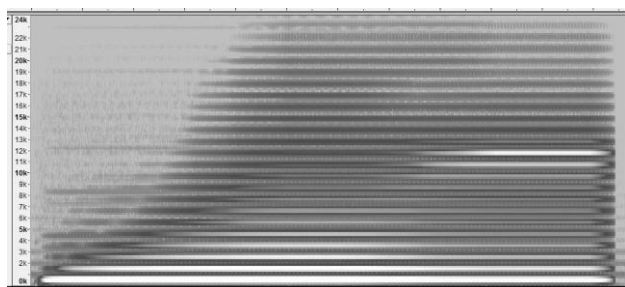


Figure 13: Filter spectrogram

## 5. CONCLUSIONS

In this paper we presented the design and implementation results of a high quality subtractive synthesizer implemented on the Arduino Due platform. We also demonstrated that it is possible to have an additive synthesis organ inspired on Hammond design on the same platform at the same time. The complete implementation is open, public and focused on low cost platforms. Although the implementation was created on the Arduino Due, we believe it fits also on any other similar board containing an ARM Cortex-M3 processor or other processors with similar architectures (maybe removing some features or modules if using a slower clock than 84 MHz).

The fixed-point approach completely met the expectations for the implemented algorithms. With the 32 bits processor, we could use fixed point calculations with a good resolution.

The required time for processing the main modules allowed us to include two oscillators, two envelope generators with selectable trigger mode, one LFO, one low pass filter, one effect module, a MIDI receiver and a keybed on a 48 kHz sample rate. The PolyBLEP algorithm, implemented with fixed point calculation also presented good results, by removing a significant part of the audible aliasing from the pulse and sawtooth waves.

Regarding the filter implementation, we used two filter designs from [5] and [14] and, although they are good starting points and bring a lot of possibilities to the sound creation, we believe this is one of the key points for further improvement. As future work, we propose the implementation of different filters that could replicate better the filters used by classical synthesizers, such as Moog, ARP or Oberheim.

The object-oriented design made it possible to easily modify the synthesizer features and allowed us to validate the algorithms by running the code on a standard PC. We encourage the usage of this approach by having function mocks or empty function calls for hardware related functions.

Finally, both subtractive and additive synthesis are available on the same code and platform, and the user can select between them with a simple toggle switch connected to an Arduino Due digital pin.

As for next steps, besides a better filter implementation, we intend to implement more effect types, “keyboard tracking” on filters and LFOs, to make the parameter control to be adjustable depending on the note being played (as present on old analog synthesizers), two note polyphony (but this would reduce the total number of modules instantiated in the system) and other waveform types, such as sample-and-hold and noise.

Furthermore, the successful implementation of two synthesizers on the simple Arduino Due board using fixed point arithmetic demonstrates that complex audio functions can be implemented cheaply in this platform and provide a good audio quality. It naturally opens up several other opportunities for implementing other forms of audio synthesis, other functions and modules, and also innovative ideas with the contribution of a larger community.

The source code is available in a public repository [16]. The schematics and some pictures of the synthesizer finished are also included in the repository. A video of the project for sound demonstration is available at <https://youtu.be/asuycIvozhg>.

## 6. References

- [1] P. Jamieson and Jeff Herdtner, “More missing the Boat - Arduino, Raspberry Pi, and small prototyping boards and engineering education needs them” in *Frontiers in Education Conference (FIE), 2015. 32614 2015. IEEE*, 2015. doi: 10.1109/FIE.2015.7344259
- [2] T. Barrass. (2016). “Mozzi Sound Synthesis Library for Arduino”. Source code available at <http://sensorium.github.io/Mozzi/>
- [3] A. Huovilainen, “Design of a Scalable Polyphony-MIDI Synthesizer for a Low Cost DSP”, M. S. thesis, Department of Signal Processing and Acoustics, Aalto University School of Science and Technology, Espoo, Finland, 2010.
- [4] L. Biddulph and J. Ziembicki. (2000). “AVRSynth”. Available at <http://www.elby-designs.com/contents/en-us/d5.html>
- [5] M. Finke. (2013). “Martin Finke’s Blog”. Available at <http://www.martin-finke.de/blog/articles>
- [6] Y. Usson, “Yusynth”. Available at <http://home.yusynth.net/>
- [7] *modular.br*, Facebook private group, containing 1.108 members on September 2016. Available at <https://www.facebook.com/groups/modularbr/>
- [8] < Omitted for blind review >
- [9] D. Crombie, *The Complete Synthesizer: A Comprehensive Guide*, Omnibus Press, 1982.
- [10] Devahari, *The Complete Guide To Synthesizers*, Prentice-Hall, Inc., 1982.
- [11] E. C. Kisenwether and W. C. Troxell, “Performance Analysis of the Numerically Controlled Oscillator” in *40th Annual Symposium on Frequency Control*, 1986. doi: 10.1109/FREQ.1986.200971
- [12] V. Valimaki and A. Huovilainen, “Antialiasing oscillators in subtractive synthesis” in *IEEE Signal Processing Magazine*, v. 24, n. 2, p. 116–125, 2007. doi: 10.1109/MSP.2007.323276
- [13] W. Pirkle, *Designing Software Synthesizer Plug-Ins in C++*, Burlington, MA: Focal Press, 2014. [E-book] Available: Safari Books Online.
- [14] P. Kellett, “Moog VCF, variation 1, 24dB resonant low pass”, source code available at <http://www.musicdsp.org/archive.php?classid=3#25>
- [15] Delsauce. (2013). “ArduinoDueHiFi - An I2S audio codec driver library for the Arduino Due board”, source code library available at <https://github.com/delsauce/ArduinoDueHiFi>
- [16] <https://github.com/rppirotti/rgsynth>

# ELSE Library for Pure Data

Porres, Alexandre Torres

EL Locus Solus (independent center)  
São Paulo, SP

el.locus.solus@gmail.com

## Abstract

The main computer music languages for Live Electronics nowadays are: Max, Pure Data (or just “Pd”) & SuperCollider. In comparison to the other two, Pd offers a very limited set of functions in its main distribution (a.k.a “Pd vanilla”), relying heavily on external libraries as add-ons.

This paper describes the ELSE external library for Pd, which brings elements that were missing in Pd and its current external libraries when compared to other computer music environments. It also includes functionalities not available elsewhere and revises elementary building blocks of computer music already found in Pd or other systems, such as in the design of oscillators.

The ELSE library provides a large collection of objects that were carefully and meticulously designed to improve the patching experience for Pd, allowing some techniques of computer music to be implemented more conveniently.

It is also part of a didactic project for computer music that uses Pd to cover a wide range of computer music techniques and DSP topics in a tutorial with over 350 examples. The final goal is to strongly depend on ELSE to patch the examples from the tutorial.

## 1. Paper Structure.

Before describing and discussing the ELSE library, the paper first contextualizes the current state of Pd in section 2, so it is made clear where and how the ELSE library fits in.

Section 3 describes the motivations and goals of the ELSE library as a counterpoint to the previous section. Section 4 describes details from the and section 5 presents the final discussion and further work.

## 2. Contextualization: The current state of Pure Data and its external libraries.

Pure Data<sup>1</sup> is a visual programming language, quite similar to Cycling 74’s Max<sup>2</sup>, which is a commercial software. One of the main differences is that Pd is free an open source. Other open source environments for computer music (such as Csound<sup>3</sup>, Chuck<sup>4</sup>, SuperCollider<sup>5</sup>) are not visual, but textual instead. This makes Pd the only of its kind (an open source visual programming language), not to mention one of the most widely used.

An indicative of the user base population of these systems may be the size of its community on Facebook. The Pure Data community<sup>6</sup> has over 11.000 members, while SuperCollider’s<sup>7</sup> has less than 5.000. Max<sup>8</sup>, despite being a commercial software, still has the biggest community, with over 14.000 users.

The visual programming paradigm may be more intuitive and comfortable for non programmers (such as artists and musicians), making it more popular and known outside the computer music niche. This may explain the great number of the user base of both Pd and Max over SuperCollider. But a big user base does not imply in a proportionally large community of developers.

Most of the people involved in the SuperCollider community are experienced programmers, so they can collaborate much more to its development. Not surprisingly, the SuperCollider community is much more active than Pd’s. Not only that, but its development is

1 Link: <https://puredata.info/>

2 Link: <https://cycling74.com/products/max>

3 Link: <http://csound.github.io/>

4 Link: <http://chuck.cs.princeton.edu/>

5 Link: <http://supercollider.github.io/>

6 Link: [www.facebook.com/groups/4729684494/](https://www.facebook.com/groups/4729684494/)

7 Link: [www.facebook.com/groups/supercollider/](https://www.facebook.com/groups/supercollider/)

8 Link: [www.facebook.com/groups/maxmspjitter/](https://www.facebook.com/groups/maxmspjitter/)

not centralized, while the development of the main distribution of Pure Data (Pd Vanilla) has always been centralized around Miller Puckette, its main author and distributor.

Pd Vanilla comes with a minimal set of objects. Miller Puckette has been developing Pd for over 20 years, in a notoriously slow pace, but as part of a conscious decision. Miller will only include a fix or a new functionality to Pd when he is certain that it is the best choice and won't need changes in the future. He aims for a canonical implementation of functions and is concerned that Pd patches will all run 30 years from now<sup>9</sup>.

This orientation ends up promoting not only the development of external libraries for Pure Data, but also other distributions – or “flavors” – of Pd. One important parallel distribution named “Pd Extended” appeared in the early 2000's. It provided, besides the basic Pd Vanilla set of objects and GUI, a few other functionalities and dozens of external libraries already included.

Pd Extended had a major role in making Pd popular, but it has unfortunately been abandoned by its main author: Hans-Christoph Steiner. Its latest version, released in 2013, is very outdated when compared to recent Pd Vanilla developments. That notwithstanding, many still use Pd Extended as it is still distributed and considered convenient.

The main feature of Pd Extended is surely its provided external libraries. Therefore, with the abandonment of Pd Extended, the Pd community decided it was best to, instead of further developing on it, make it easier to install the external libraries from Pd Extended into Pd Vanilla. As a result, Pd now has an external installer since version 0.47-0. Although this did partially compensate the abandonment of the Pd Extended distribution, there is a bigger underlining issue, which is the fact that the vast majority of the external libraries available in Pd Extended hasn't had a maintainer or developer for a long time.

Thus, even before Pd Extended's last release, most of its main features and

differentials had been long abandoned. Furthermore, there was not a big criteria for including libraries into Pd Extended. This means that some external libraries were included still in an experimental or early stage of development and were never further developed. The outcome is that you can easily find bugs and badly documented externals. So fixing, maintaining and developing for such libraries would be more important than bringing Pd Extended back to life.

However, so to speak, Pd-Extended has recently reincarnated as “Pd-L2ork 2.0 / Purr Data”[1], which started as a fork of Pd Extended in 2009 and became the Pd-L2ork distribution (but only for Linux). The Pd-L2ork 2.0 version is cross platform and was released in early 2017 by the name “Purr Data”. It does have all the libraries and features from Pd Extended plus more of its own. But the issue of including mostly abandoned libraries it inherited from Pd Extended still remains.

### 3. ELSE's motivation and goal

Pd Vanilla offers a minimal package for computer music but has many external libraries that can be easily installed via its external manager, not to mention the Purr Data distribution, which provides several external objects already built-in. As follows, one could reason that Pd operates in a modular structure, relying on declaring extra packages that may be imported when needed. In this way, external libraries should compensate the lack of essential features in Pure Data Vanilla, not leaving it behind other environments.

However, that is not what we have in practice. Besides the fact that most libraries are simply abandoned nowadays, the collection of external libraries in Pd Extended presents a big kludge instead of an organized set of dedicated external packages. Most are just relatively small random sets of functions, which often carry virtually the same functionality of other externals. Hence, Pd's externals libraries needed a major overhaul and revision in order for it to claim an actual modular structure.

From the few libraries in Pd Extended that are still in active development, the main and largest one is Cyclone. Though it did spend about a decade with no significant development,

9 As stated in this video from the Pd Conference 2016, about the future of Pd. Specially from 53:52 up to 59:30 Link: <https://www.youtube.com/watch?v=cAWFk1PPTtk&feature=em-lss>

it went through new development phases recently, and it's under the maintenance of Alexandre Torres Porres, Derek Kwan and Matt Barber since february 2016 [2].

Cyclone clones MAX/MSP to Pure Data, being popular for providing some level of compatibility between the two environments. Its latest version (cyclone 0.3, still in the beta phase) has about 200 objects and should be merged to Purr Data soon.

In this context, the ELSE library aspires to provide a substitute for many abandoned Pd libraries, in a quite ambitious goal to centralize many essential elements into a single library, getting rid of the need to search amongst several libraries for a simple and essential object.

The initial motivation of the ELSE library was actually to include important functions that were not available in Pd Extended/Purr Data libraries. This need emerged from an educational project, which is the development of a tutorial for computer music based on Pure Data by the author<sup>10</sup>, which now has over 350 examples and covers a wide range of computer music techniques and DSP topics [3-4].

The author has been developing this original tutorial to teach computer music for 9 years, up to a point where all the relevant existing objects in Pd Extended were used up, so there was a need for something “else”.

A parallel idea was to clone the UGENs from SuperCollider, creating a dedicated library of clones such as Cyclone. But this was shortly abandoned as it would constrain to the same architecture as the original UGENs, and soon it was decided that these functions should have different design choices. One reason is the consideration that some of the original design choices were not good, and the other one is that there would have to be adaptations to the context of the Pd environment anyway. therefore, this idea was merged into the ELSE project, although a good name for a SuperCollider clone library was already thought of (SuperClonider).

A further expansion of the original scope – as already mentioned – includes the revision of many externals from Pd Extended. This

decision was motivated by the fact that such externals have been abandoned and could benefit from revisions into a new library, so the didactic material by the author would rely mostly on externals maintained by himself.

Now, since the author is also a main collaborator of Cyclone<sup>11</sup>, the idea is that the didactic material will rely heavily on both of these libraries. But for the great majority of other externals from Pd Extended used in the examples, the ultimate goal is to replace them all with a newly written external from ELSE.

It needs to be noted that even functions from objects already available in Cyclone or Pd Vanilla are also being revised by objects in ELSE, such as the case with oscillators, described in the next section.

## 4. The ELSE library

This section describes some of the features and objects from the ELSE library. At the time of its writing, the project is still in an alpha stage and counts over 110 objects, but a first beta release is planned to be made available by the time of the SBCM conference<sup>12</sup>. The following subsections describe important objects, groups of objects, and features of the ELSE library.

The name of the library stands for *EL Locus Solus' Externals*, where “EL Locus Solus” is the independent center run by the author. It functions as a production agency and a school that offers computer music courses<sup>13</sup>.

### 4.1 Oscillators

The oscillators available in ELSE bring together functionalities not available elsewhere. Not only in other Pd externals, Max's and SuperCollider's internal functions, but also in other computer music languages such as Csound and Chuck and even in a modular environment like Reaktor<sup>14</sup>. Namely, they offer

11 The latest releases of Cyclone are available at: <https://github.com/porres/pd-cyclone>

12 The latest releases of ELSE are available at: <https://github.com/porres/pd-else>

13 Link: <http://alexandre-torres.wixsite.com/el-locus-solus>

14 Find Reaktor at <https://www.native-instruments.com/en/products/komplete/synths/reaktor-6/>

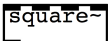


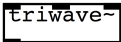



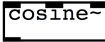

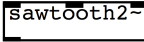
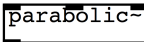
10 Available for download at: <https://github.com/porres/LiveElectronicsTutorial>

hard sync, phase modulation and accept negative frequencies.

The basic oscillators in Pure Data (such as [osc~]) have a secondary control inlet for resetting the phase. If it were a signal inlet, a signal input could trigger the oscillator to perfectly reset in sync to another frequency (i.e. hard sync technique).

The oscillators in ELSE follow the usual design from vanilla objects, in which the first inlet controls the frequency and the first secondary inlet resets the phase. The phase can be reseted in the same way as with [osc~], with a control message, but it also works with a signal impulse with a different logic.

Having a separate action for both signal input (sync at impulses) and control messages poses a challenge. It is not only unusual in Pd but it's also very tricky to implement it in the Pd API. For that, the author applied a technique developed by Matt Barber for Cyclone, nicknamed "magic" [2].

	Square oscillator with pulse width
	Pulse oscillator with pulse width
	Variable saw-triangle oscillator
	Triangle oscillator
	Single sided impulse oscillator
	Double sided impulse oscillator
	Sine oscillator
	Cosine oscillator
	Sawtooth oscillator
	Another sawtooth oscillator
	Parabolic oscillator

**Figure 1: Oscillators in ELSE**

## 4.2 Sample accuracy and impulses

Many Pure Data audio objects are unable to be triggered with sample precision/accuracy. The same can be said about Max, though to a

lesser extent. Dealing with sample accurate processes in both Max and Pd is tricky, being more common to use control messages (either general control messages or bangs) instead. This is surely more efficient computationally, but some computer techniques depend on controlling parameters at audio rate – such as “hard sync”.

Implementing “hard sync” in Pd is not impossible, but it's just not elegant, as it is a workaround that could be avoided if only the oscillators were designed to accept a sample accurate phase reset control.

Objects from the ELSE library are strongly aimed to sample accurate processes, for which an impulse oscillator is very important. Objects can then be programmed to respond when a trigger signal is different than 0, or when they transition from 0 to a positive value, which is exactly what an impulse oscillator provides at a fixed rate, so it can be thought of as an audio rate metronome.

Since Pd Vanilla was not well planned for sample accurate processes, it does not have an impulse oscillator. Max's [train~] (also available in Cyclone) is the closest thing to it, but not an actual oscillator.

SuperCollider is more oriented towards sample accurate processes, and thus has an Impulse oscillator UGEN, but it does not accept negative frequencies. Pd Extended/Purr Data has [impulse~], an impulse oscillator from the sigpack library, but it has bugs and also does not accept negative frequencies.

Therefore, [impulse~] is one of the objects already available in the Pd ecosystem that needed a revision inside ELSE. It is one of the few that has the same name as a pre existing external. But in such cases, the provided externals in ELSE are fully backwards compatible to the existing ones.

Thus, [impulse~] from ELSE solves the existing bugs in [impulse~] from sigpack and provides more features (negative frequencies, phase modulation and hard sync) – note that the [impulse~] object in ELSE can also be instantiated as [imp~], for short, as in figure 1.

Following this sample accurate structure, many objects in ELSE are triggered by impulses, such as the oscillators' hard sync inlet, or negative to positive transitions. Not



only that, but other objects in ELSE also generate impulses as result of signal analysis, such as [zerocross~] (when detecting zero crossings), [changed~] (when detecting signal change) and more, see figure 2 below.

<code>changed~</code>	Detect signal changes
<code>changed2~</code>	Detect direction changes
<code>elapsed~</code>	Elapsed number of samples
<code>togedge~</code>	Detect 0 to non-0 transitions
<code>zerocross~</code>	Detect zero crossings
<code>dust~</code>	Random positive Impulses
<code>dust2~</code>	Random impulses
<code>pulsecount~</code>	Count pulses/triggers
<code>pulsediv~</code>	Pulse divider
<code>sh~</code>	Sample and hold
<code>toggleff~</code>	Toggle flip flop
<code>trigate~</code>	Trigger and timed gate
<code>ramp~</code>	Resettable ramp

**Figure 2: Objects that either respond to or generate impulses.**

### 4.3 [sh~] and [downsample~]

The [sh~] object from ELSE is a *sample and hold* module that is triggered by impulses. Any sample and hold object needs a trigger signal input to make it “sample an input value and then hold it”. Therefore, it is inherently a sample accurate process.

Pd Vanilla has its own sample and hold object, named [samphold~]. But since Pd Vanilla isn’t aimed towards sample accurate processes and lacks an impulse generator, [samphold~] was designed to be triggered with [phasor~].

This is not a conventional design, but a workaround in order to adapt a sample and hold unit in an environment that lacks impulse trigger signals. The way [samphold~] works is

that it is triggered whenever its right inlet input decreases in value. This happens with [phasor~] at period transitions, allowing it to trigger [samphold~] at that moment, but only when it has a positive frequency smaller than the sample rate.

The most critic issue is that [phasor~] is useless to trigger [samphold~] when it is running with negative frequencies, since what happens in negative frequencies is that [phasor~] inverts its direction and outputs a downwards ramp instead. Therefore, apart from the period transitions, it always outputs a value smaller than the previous one – exactly what triggers [samphold~], so it can’t work that way.

Max’s sample and hold object is [sah~], also available in Pd via Cyclone. It has a more common design, allowing it to be triggered with an impulse oscillator – since it is triggered when the signal raises above a given threshold. The [sah~] object also requires that the signal falls below the threshold so it can be triggered again. Therefore, the highest rate [sah~] can be triggered is the nyquist frequency.

The [sh~] object from ELSE is similar to [sah~], but it may also not require the signal to fall below the threshold value (this is actually done by default). This allows it to be triggered with an impulse oscillator up to the sample rate frequency. As such, it may also act as a “gate or hold”, where instead of a trigger signal, you have a gate signal that allows the input signal continuously through when it is higher than the specified threshold. This is, in fact, similar to SuperCollider’s Gate UGEN.

The [sh~] object combines features from SuperCollider’s Gate, Max’s [sah~] and Pd’s [samphold~], making it the most complete and versatile sample and hold object design. The conjunction of ELSE’s [imp~] and [sh~] objects also has more potential than previously available.

One possible application is downsampling, where it can go up to the sample rate frequency (positive or negative). But the [downsample~] object from ELSE already does this for you. One can think of it as a sample and hold object that only needs to receive a trigger rate in hertz (negative frequencies also accepted). So you’d only need to implement downsampling with [imp~] and [sh~] if you

wish to use other features from these two objects. One difference, though, is that [downsample~] can also interpolate between values.

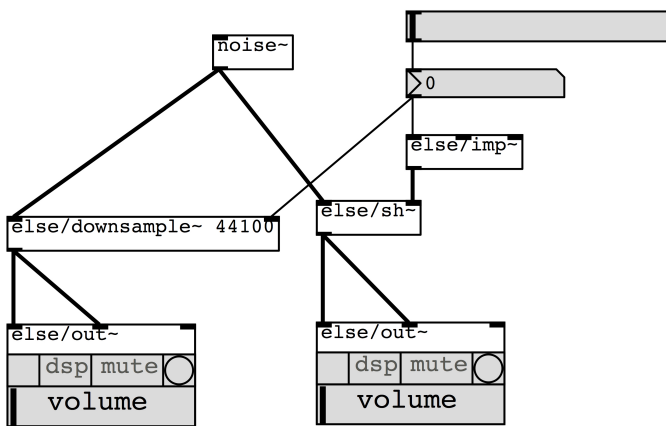


Figure 3: [downsample~] and an equivalent patch with [sh~] and [imp~]

#### 4.4 [pimp~]

The [pimp~] object from ELSE is a combination of Pd Vanilla's [phasor~] and ELSE's [imp~] (or [impulse~]). It's basically [imp~] with an extra phase output in the left outlet. Conversely, it can be thought as a [phasor~] with an extra impulse output in the right outlet.

The impulse happens at every phase period transition (in both negative or positive frequencies/directions). Like [imp~], [pimp~] also has inlets for phase modulation and sync.

[pimp~] is very convenient for both driving a process with its phase output (such as reading a sample or envelope wavetable) and also triggering other objects at period transitions.

The need for such an object was found when dealing with granulation patches. For instance, the *B13.sampler.overlap* example, from the audio examples from Miller Puckette's book [5] (which are available in the Pd Vanilla distribution) uses [phasor~] to generate overlapping envelopes and also trigger [samphold~] objects, which samples new parameter values at the end of the envelope.

This is the basis of other granular based patches such as the time stretch/compress with independent pitch shifting from the next

example in the series (*B14.sampler.rockafella*).

But this is only possible if you're reading the table/sample in the original direction. The patch does not work if you're reading the table backwards, as it would require the [phasor~] to run at a negative frequency, and [samphold~] simply does not work that way, as already explained.

The workaround here would be to analyze the signal from [phasor~] and generate an impulse signal when transitioning from one period to the next, in both positive or negative frequencies. But even so, that would require a Pd external such as [sah~] for the sample and hold function, since [samphold~] cannot be triggered like that.

The way you can generate impulses from [phasor~]'s period transitions, whether its frequency is positive or not, is by realizing that whenever it reaches the end of its period, there is a big leap in value. So you can measure the absolute difference between the current and the last value, and we can capture every period transitions up to the nyquist frequency (positive or negative) whenever it is higher than or equal to 0.5.

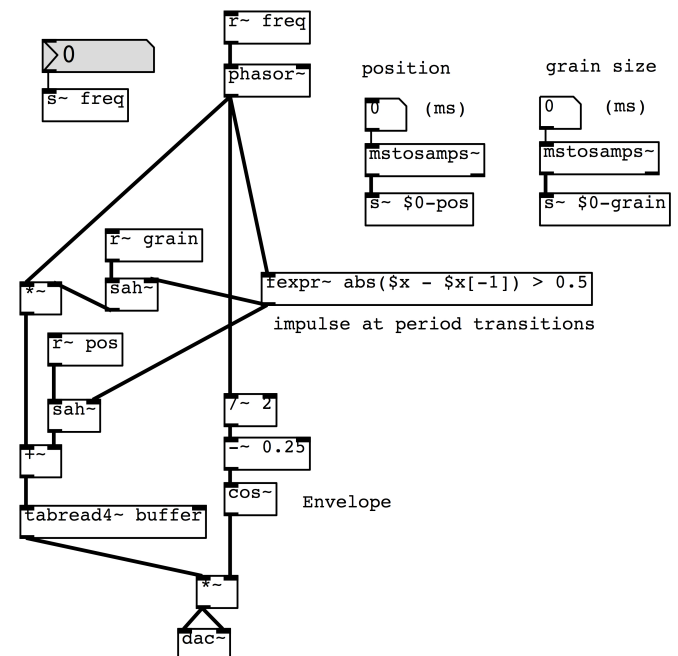
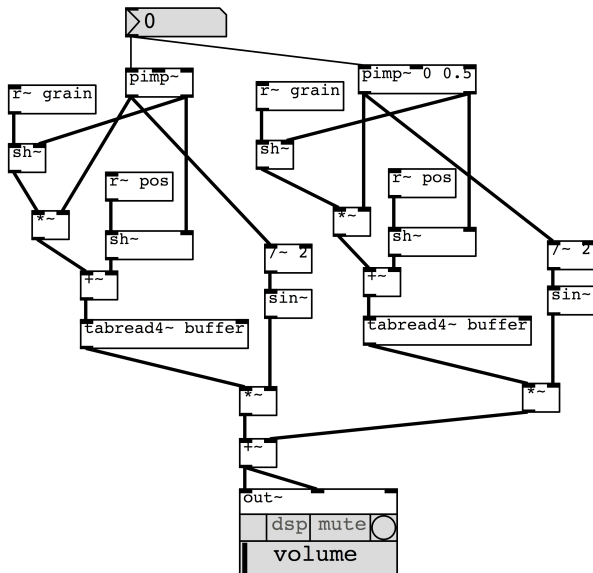


Figure 4: Example of granulation by getting an impulse from period transitions of a [phasor~] and relying on the [sah~] external.

This is not unreasonably hard to implement in Pd and can be done just with one [fexpr~] object (see figure 4 above), but it's just not elegant. Again, all these workarounds could be avoided if simply the design choices of the objects were better.



**Figure 5: a simplification of the granular patch by adopting objects from ELSE such as [pimp~]**

Objects from the ELSE library previously mentioned can simplify this, such as [imp~] in conjunction with [sh~] or just [downsample~]. But [pimp~] is also provided as a single and more elegant solution for this particular purpose. Check the adaptation in figure 5, and how the patch is now much cleaner and also able to include two overlapping grains into a more concise space.

#### 4.5 Chaotic and Stochastic generators

All of the chaotic/stochastic generators in SuperCollider are being cloned. For now, these are: [brown~], [clipnoise~], [crackle~], [cusp~], [gbman~], [henon~], [lincong~], [logistic~], [latoocarfian~], [quad~], [standard~]. Apart from those objects available in SuperCollider, the Ikeda attractor has also been implemented as an extra generator (named [ikeda~]).

The [lfnoise~] object is a low frequency chaotic generator. There are similar objects to [lfnoise~] for Pd, but with less functionalities. In Max's (and Cyclone) you have [rand~]. The

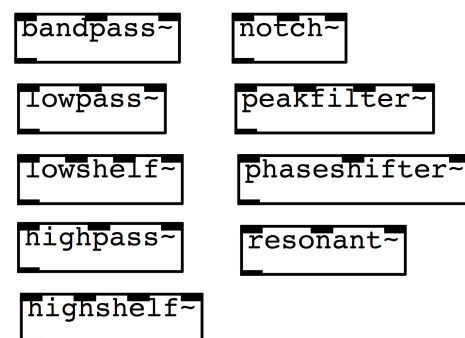
zexy library for Pd has [noish~] and [noisi~]. SuperCollider has the LFNoise generators. But [lfnoise~] differs for accepting negative frequencies and allowing sync with impulses, following the same structure of ELSE's oscillators. Else also provides [random~], which generates random signal values when receiving a trigger signal.

#### 4.6 Filters

Pd Vanilla lack many basic filters, relying mostly on [biquad~] to generate general second order filters. But even so, it lacks a biquad coefficient generator and only accepts control input for the coefficients.

The ggee library for Pd offers several generators for [biquad~] – covering elementary filters such as lowpass, lowshelf, bandpass, and so on – but the parameters are still constrained to control messages, not allowing signal rates for simple filter sweep patches with a LFO.

Max also relies on a [biquad~] object, but at least it has signal rate input and offers a biquad coefficient generator with signal rate inlets: [filtercoeff~]. Both have been recently cloned into Cyclone, but it was considered necessary to create simpler and friendlier solutions for such elementary filters in ELSE, which would be single objects for each filter type.



**Figure 6: second order filters from ELSE**

Most of such filters are made available in SuperCollider's BEQSuite group of filters as single classes. The difference in ELSE's objects is that, instead of a reciprocal Q, it has a more intuitive and friendly design to allow the resonance parameter to be set in either Q or bandwidth in octaves. The filter formulas are

from Robert Bristow-Johnson's work<sup>15</sup>.

## 5. Final Discussion and further work

This paper briefly presents and describes the ELSE library for Pure Data. The development of this library follows an experience of over a decade in patching with Pd Extended, and about 9 years of developing didactic material based on Pd Extended patches.

More recently, the author has also studied and used SuperCollider and Max to teach computer music. Throughout the course of these years of patching and teaching a wide range of computer music techniques with the existing tools in these three environments, the author accumulated many issues and frustrations with the design of the available building blocks.

A discussion and details behind the creation of some of the externals is given in this paper. The discussion could be further detailed in order to describe every decision behind the design of all the objects in the library, but the given examples make the general purpose clear, which is the need found by experience to improve some elementary building blocks of computer music and include new functionalities that were missing.

Given the current state of Pure Data, with many of the existing libraries abandoned, the development of this library also aims to replace the need of many of the existing externals.

The author has also discussed the inclusion of ELSE in Purr Data as a newly available library as soon as a more stable release of ELSE is made available, and after Cyclone is first included there.

The ELSE library has a didactic motivation. Its ultimate goal is to be the applied in the development of computer music examples from the didactic material developed by the author, alongside the Cyclone library, also under the maintenance of the author and already being included in Purr Data soon in its latest state.

Currently, over 350 examples have been developed making extensive usage of available objects in Pd Extended. As soon as a first

version of the ELSE library is made available and included in Purr Data, the next step is to adapt this didactic work to Purr Data and start relying on existing externals from ELSE more and more.

Short term plans for ELSE include revising the documentation and adding band limited functionalities for the oscillators. As soon as this is finished, a final 1.0 version may be released.

## Acknowledgements

Flávio Luiz Schiavoni, for being the first to help me get into the art of coding externals for Pd and for coding the first version of [median~], the first object from ELSE to be coded. Derek Kwan and Matt Barber, my fellow collaborators in Cyclone, that also teach me so much and help me a lot. Ivica Bukvic and Jonathan Wilkes for being open to accept my ELSE library in Purr Data as soon as it is stable and ready for it. Miller Puckette, for being awesome and being the giant who created Pure Data and Max, plus all the others in the Pd Community, who are always very helpful.

## References

- [1] I. Bukvic, J. Wilkes and A. Graef, *Latest developments with Pd-L2Ork and its Development Branch Purr-Data*, PdCon16, NYU, 2016.
- [2] A. Porres, D. Kwan and M. Barber, *Cloning Max/MSP Objects: a Proposal for the Upgrade of Cyclone*, PdCon16, NYU, 2016.
- [3] A. Porres, *Teaching Pd and using it to teach: yet another didactic material*, PdCon09, São Paulo, 2009.
- [4] A. Porres, *Patches de Pd para Computação Musical*, Revista Vórtex, Curitiba, v.2, n.2, 2014, p.198-200.
- [5] M. Puckette *The Theory and Technique of Electronic Music*. World Scientific Press, 2006.

15 Cookbook formulae for audio equalizer biquad filter coefficients, by Robert Bristow-Johnson, available in <<http://shepazu.github.io/Audio-EQ-Cookbook/audio-eq-cookbook.html>>.

# Gestures of Body Joints, Musical Pulses and Laban Effort Actions: Towards an Interactive Tool for Music and Dance

Leandro Souza<sup>1</sup>, Sérgio Freire<sup>1</sup>

<sup>1</sup>Laboratory for Performance with Interactive Systems – School of Music - UFMG  
Av. Antônio Carlos, 6627 - Campus Pampulha – 31270-901 Belo Horizonte, MG

lleandro\_ssouza@hotmail.com, sfreire@musica.ufmg.br

## Abstract

In this paper we present a proposal of segmentation and description of gestures of dancers based on the Laban Movement Analysis. This procedure is the main core of an interactive tool for music and dance, implemented in Max/Msp/Jitter and using Kinect, which aims to associate corporal and sonic gestures. The segmentation is done by means of the inspection of the zero-crossings of the acceleration curve of each body joint. After that, different descriptors for each gesture are extracted, and they feed routines to estimate three of the Laban effort factors: time, space and weight. From these data, it is possible to classify the gestures according to the eight basic Laban effort actions. A case study is also presented, in which we search for correlations between the pulse and character of a musical excerpt and the rhythms and qualities of the extracted gestures.

## 1. Introduction

Recent technologies are responsible for the increasing number of researches and activities dealing with music and movement, a relationship deeply rooted in every human culture. Our research is located at the intersection of three areas: study of the sound gesture, study of the gesture in the context of new sound interfaces and the study of the movement qualities in dance. We propose strategies and tools for the development of interactive musical situations that explore dance and music within the context of electroacoustic composition, live electronic performances and other interactive performances. For this, we started with the investigation of gestures in both artistic modalities, using spectromorphological ideas

for the music and motion capture techniques for the dance. We aim to develop strategies of interaction based on the gestural qualities extracted from both domains. In this article, we will focus on the analysis of the gesture in dance, where digital techniques of motion capture, associated with elements from Laban Movement Analysis, are explored with the aim of developing methods of realtime segmentation and description of gestures in dance. We also present a case study: a dancer improvisation based on some excerpts from the ballet *Petrushka*, by Igor Stravinsky, where we analyze the correlations between musical pulses and characters, body rhythms and the movement qualities.

## 2. Segmentation and Description of Body Movements

The interaction between sound and body gestures is a fundamental concept in our research. We have chosen to use the term *gesture*, in agreement with Jensenius et al [1], for we also believe that the "notion of gesture somehow blurs the distinction between movement and meaning." Or, as stated by Schacher, "a gesture is a sequence of movements that form a whole, a gestalt and can be recognized as a semantic unit" [2].

Thus, in extracting and analyzing information from the movements, we are looking for qualitative aspects, and for this we start from theories that aim to clarify the expressive aspects of the movement, such as the Laban Movement Analysis. LMA, as it is known internationally, is a methodology for observing, describing, interpreting and recording human movement and was developed by Rudolf Laban (1897-1958) in the middle of the 20th century. LMA has been em-

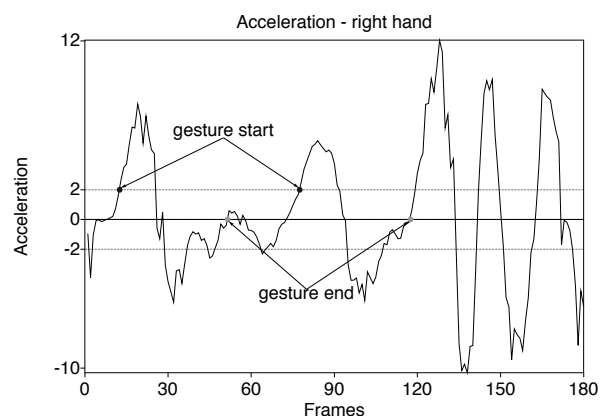
ployed in several areas such as dance, performing arts, sports, physical therapies, psychology and behavioral sciences. Its methodology is formed by four main categories: Body, Effort, Form and Space. In every movement, all four categories are present, but with different emphasis. For our work, the category of Effort is the most significant, since, according to Laban[3], it is possible to analyze the expressive possibilities of the body in movement through the dynamic qualities of the Effort. According to the Body category, the body can be separated into parts, and each moving part is susceptible of an Effort analysis. Thereby, in our approach, the movement of different joints - digitally captured - have gestural capacities, because they can perform movements with expressive qualities.

Laban proposed four factors of Effort: **a)** Flow - related to the degree of movement control; **b)** Space - related to the trajectory of movement in space; **c)** Weight - related to the resistance to gravity of the movement; **d)** Time - related to the duration of the movement. Each factor can oscillate between two extremes. The flow factor sways between contained and free, the space factor between direct and indirect, the weight factor between strong and light, and the time factor between sustained and sudden. Hence there are two polarities: (1) *Indulging*, with free Flow, indirect Space, light Weight and sustained Time; (2) *Condensing*, with contained Flow, direct Space, strong Weight and sudden Time. Inspired and based on the LMA effort category, we have developed and implemented in Max-Msp-Jitter a real-time tool for the analysis of dancers movements. We found works that adopt a segmentation procedure based on temporal windows, whose lengths may be adjusted manually or automatically [4, 5]. Nevertheless, we opted for segmentation from the positional data of the parts and joints of the body, by delimiting each segment from the observation of the zero-crossings of the acceleration curves. This approach was successfully employed in other works of motion analysis [6, 7, 8].

## 2.1. Realtime implementation

We extract the 3D position data generated by Kinect for each of the joints of the body. This

data is formatted into OSC<sup>1</sup> messages by the Synapse<sup>2</sup> software, and sent to the Max-Msp-Jitter programming environment. Every joint is assigned with a label, which may be accessed as a variable within the implemented patches and subpatches. The reference axes for the torso data are given by the position of the Kinect (called the world reference); the remaining joints use the torso coordinates as reference (called the body reference). From these data, we estimate the scalar velocity of each joint by calculating the Euclidian distance between two consecutive points (its first derivative). The acceleration curve is estimated by the second derivative. The capture rate is 30 frames per second. In order to smooth out spatial and temporal irregularities of the device, we apply a moving average filtering to this data: we use a 2-point filter for the displacement curves, and a 7-point filter for the velocity and acceleration curves. The beginning of each gesture is determined when the acceleration curve exceeds a positive limit (which is adjustable for each joint), and the end of the gesture is defined when the curve returns to the zero value after having reached a negative limit (Figure 1). The speed and acceleration values are calculated with regard to the sample rate, instead of the common unities for time and length.



**Figure 1: Segmentation of the Right Hand Acceleration Curve with a threshold value 2.**

For each segmented gesture we estimate the following descriptors: **a)** *dur* - duration of the gesture in *ms*, or the time elapsed between its be-

<sup>1</sup>Open Sound Control (<http://opensoundcontrol.org/>)

<sup>2</sup><http://synapsekinect.tumblr.com/>

ginning and end; **b)** the total displacement in *mm* made by the joint, which is represented by the integration of the Euclidian distances between every adjacent point; **c)** the displacement modulus, which is the length of the line segment between the start and end points; **d)** the mean speed, calculated as  $b/a$ ; **e)**  $l\_ratio$  - the ratio between  $b$  and  $c$ ; **f)**  $accel$  - the mean value of the absolute values of every point in the acceleration curve; **g)**  $a\_ratio$  - the absolute value of the ratio between the positive and negative acceleration mean values; **h)** direction of the gesture on x-axis: *left* or *right*. It is derived from the difference between the start and end x-values; **i)** direction of the gesture on the y-axis: *up* or *down*. Calculated as in  $h$ ; **j)** direction of the gesture on the z-axis: *front* or *back*. Besides, we also extracted the curve of the torso's floor speed, and other qualifiers based on [9] and [10]: a contraction/expansion index and the distance between different joints, which will not be discussed in this paper.

Figure 2 depicts the flowchart of the implemented segmentation procedure.

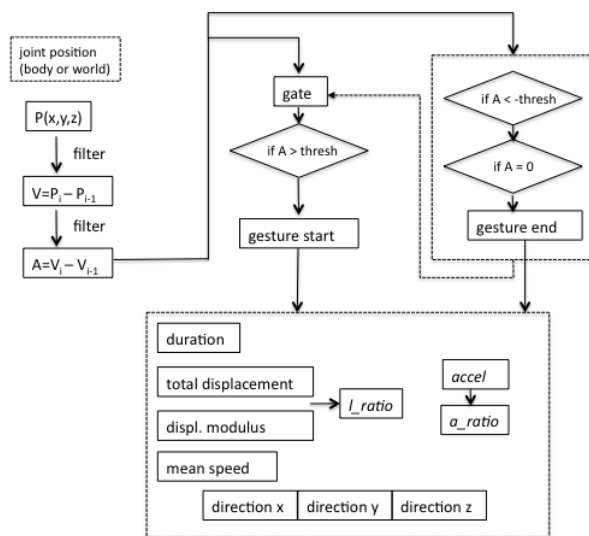


Figure 2: Flowchart of Gesture Segmentation

## 2.2. Estimating Laban Effort Factors

The combination of these descriptors may help to classify the gestures according to the LMA Efforts factors. We have implemented algorithms to estimate three of them: space, weight and time. We are still working on a good strategy to cope with the flow factor, for it may be

applied either to one or to a possibly large set of gestures. As the effort factors are dynamic qualities, it may occur movements in which one or more factors don't have an expressive emphasis, and thus remain latent. Laban [3] considers such movements as incomplete efforts. Based on this conception, we also found useful to define a neutral region between the two poles of each factor.

**Factor Space:** this factor is directly related to the descriptor  $l\_ratio$ . The more it tends to the value 1, the more the gesture is considered direct. Practical values for the low and high limits of the neutral zone must be defined after much observation, since there is a lot of variation among the different body joints (and also among different individuals).

Condition	Result
$l\_ratio < \text{low limit}$	direct
$\text{low limit} < l\_ratio < \text{high limit}$	neutral
$l\_ratio > \text{high limit}$	indirect

Table 1: Conditions for the estimation of the Space Effort factor.

**Factor Weight:** this factor is calculated by means of a logical combination of three descriptors, organized in two conditional queries. Our assumptions are that a strong gesture should oppose gravity (an ascending movement should have more positive acceleration, and a descending one more negative acceleration), and also that a strong gesture should spend more kinetic energy (related to higher acceleration and force values) than a lighter one. Although this algorithm is unable to incorporate important aspects of Laban's weight concept, like the static forces present in very slow or minimal movements, it may contribute to the qualification of faster and wider gestures. The strength factors for each joint were also defined heuristically.

Condition 1: ( $a\_ratio > 1$  and direction *up*) or ( $a\_ratio < 1$  and direction *down*).

Condition 2:  $\overline{accel} > \text{strength factor}$ .

**Factor Time:** we use two descriptors to estimate this gesture quality: the duration of the gesture and the mean value of absolute acceleration. Sudden gestures tend to be not only short but



Condition 1	Condition 2	Result
yes	yes	strong
yes	no	neutral
no	yes	neutral
no	no	light

**Table 2: Conditions for the estimation of the Weight Effort factor.**

also to spend more energy than sustained ones. Once more, useful values for the duration threshold and for the strength factor must be defined after observation and analysis.

$dur < \text{thresh.}$	$\overline{accel} > \text{str. factor}$	Result
yes	yes	sudden
yes	no	neutral
no	yes	neutral
no	no	sustained

**Table 3: Conditions for the estimation of the Time Effort factor.**

The LMA defines eight basic effort actions, as depicted in Table 4, which are a combination of the three effort factors just described, excepting the flow (for isolated gestures, the flow factor is strongly correlated with the weight factor).

action	space factor	weight factor	time factor
punch	direct	strong	sudden
dab	direct	light	sudden
press	direct	strong	sustained
glide	direct	light	sustained
slash	indirect	strong	sudden
flick	indirect	light	sudden
wring	indirect	strong	sustained
float	indirect	light	sustained

**Table 4: The Eight Basic Effort Actions in LMA.**

### 3. Case Study: Improvising on Excerpts From the Ballet Petrushka

In order to get some insight into the relationships between the pulse and character of the music and the rhythms and qualities of the gestures of a dancer's body, we have chosen some short

excerpts from Stravinsky's *Petrushka* – none of them exceeding 15 s–, which present a clear pulse and also rhythmic and orchestral diversity<sup>3</sup>, as depicted in Table 5. A female dancer – an undergraduate student at our university – was asked to improvise freely on each of the excerpts, not long after getting acquainted with them. She was aware of the limitations imposed by Kinect, such as distance, rotation and planes constraints. As mentioned above, we the software Synapse to capture the 3D position data for 15 joints: head, neck, torso, left and right shoulders, elbows, hands, hips, knees and feet. Each rendition was also registered synchronously in audio and video<sup>4</sup>. An excerpt (*strav4*) had its data corrupted and could not be used.

excerpt	rehearsal number	pulse: on score / measured (BPM)
strav0	105	84 / 82
strav1	72	60 / 64
strav2	13	100 / 95
strav3	29	138 / 135
strav5	69	116 / 112
strav6	100	69 / 68
strav7	110-111	112 / 108
strav8	50	76 / 70
strav9	96	138 / 120

**Table 5: Chosen excerpts from Stravinsky's *Petrushka*.**

#### 3.1. Results and Discussion

We analyzed the data generated by the procedures of gesture segmentation and qualification in three complementary strategies: a general view of the articulation of gestures with regard to the musical pulses; a quantitative summary of the basic effort actions in each of the renditions; a qualitative approach of the relationship between the musical pulses and the basic effort actions. We also added a neutral basic effort action, defined as a gesture presenting the neutral quality in all three factors.

<sup>3</sup>The rehearsal numbers were taken from the 1912 orchestral score[11]. We used a recording made by the Cleveland Orchestra[12], from which we measured the average pulse.

<sup>4</sup>The videos can be downloaded at <http://www.musica.ufmg.br/sfreire/wordpress/>.



During (or after) the process of segmentation, it is possible to plot the initial moment and the duration of each gesture in relation to the pulse of the musical excerpt. This display helps to get a general view of the dancer's strategy for the improvisation, as depicted in Figure 3. We have chosen to discuss the excerpt *strav7*, because it has an interesting rhythmic structure, beginning with upbeat and finishing with downbeat chords. The pulses of excerpt have been annotated by manual means (pressing the space tab), intending to approach the dancer's perception of pulse. Initially, we can observe a tendency for starting a gesture following the pulses, as if occupying the silent downbeat: lower limbs (feet and knees) on the fourth and sixth pulses, upper limb gestures (elbows and hands) on the fifth and seventh pulses. Then we observe that the gestures tend to start more freely around the pulses and, at the end –when sound and pulses come together– we observe a search for synchrony expressed by several joints.

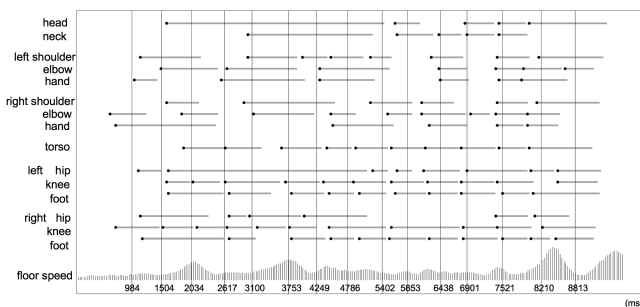


Figure 3: Segmentation of *strav7*.

The extraction of the basic effort actions presented instances of only four actions: punch, glide, slash and float. This means that every strong action is also sudden, and also that every light action is also sustained. Any of them may be direct or indirect. Table 6 presents the amount of each action detected in each rendition. Note that the dancer performed two different renditions of the sixth excerpt.

Despite this marked filtering process (rejecting 70–85% of the total), there still remains many simultaneous (or quasi-) gestures. On average, the number of neutral actions equals that of extreme actions (punch and float), although there exists significant individual deviations. The music in the excerpt *strav8* presents only fast ges-

rendition	total	P	G	S	F	N
strav0	116	6	1	8	9	16
strav1	95	11	3	1	5	21
strav2	110	9	5	5	6	7
strav3	62	3	1	7	7	4
strav5	113	8	1	9	11	13
strav6a	101	4	5	2	3	15
strav6b	124	6	8	3	7	15
strav7	124	13	3	9	7	13
strav8	97	5	7	3	10	13
strav9	132	14	3	6	6	28

Table 6: Effort Actions Detected in Each Rendition: (P)pulse, (G)lide, (S)lash, (F)loat, (N)utral.

tures on the piano solo, and was performed by the dancer with a majority of light (float and glide) actions. On the other side, *strav7* and *strav9* have a very rhythmic character, and were performed with a majority of strong (punch and slash) gestures. Excerpt *strav6a* presented the smallest percentage of all effort actions taken together (13.8%), and the second highest percentage of neutral gestures (14.8%). This excerpt has a slow pulse and explores very low and very high sounds. *Strav5* has a fast pulse, where trumpet and snare-drum have prominence, and was performed with a good amount of indirect and strong actions. Only in *strav6b* we could observe a steady tendency of starting (or ending) an action around the pulse, as Figure 4 shows. We did not find any strict correlation between the start or end points of the actions (taken individually or as a whole) and the musical pulses.

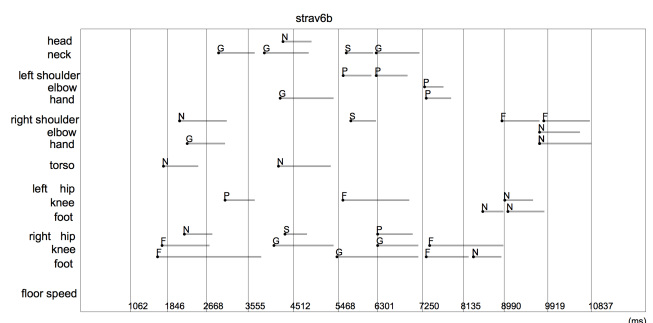


Figure 4: Effort Actions and Musical Pulses in excerpt *strav6b*.

#### 4. Final Remarks

Although our experiment was held with just one individual, it was essential for the development and implementation of tools for extraction and description of body gestures. Next, we must extend this experience to different dancers, and also to request the performance of strong/sustained and light/sudden actions for finer adjustments. As expected, it is very difficult to systematize the relations between musical pulses and the rhythms performed by the joints. Nevertheless, charts like the ones in Figures 3 and 4 may offer a general view of the dancer's strategies for the choreography. Future work points to the association of body gestures with sonic typologies, in interactive situations where dance may control the choice and the timing of sounds. In the mapping strategies, it may not be forgotten that the qualification of a gesture is done only at its end, and that it may be necessary to take in account the immediate past actions. Finally, it must be said that we are not dealing with the desired degree of precision with the setup in use. On one side, the easy access to the required equipment may be an advantage for live performances, which are the main goal of our research; on the other side, the same algorithms could be adapted to more precise setups, when available.

#### 5. Acknowledgments

We thank the funding agencies CNPq, Capes and Fapemig for their financial support for this research project.

#### References

- [1] A. Jensenius and M. Wanderley et al. Musical gestures: Concepts and methods in research. In R. Godoy and M. Leman, editors, *Musical Gestures: Sound, Movement and Meaning*, pages 12–35. Routledge, 2010.
- [2] C. Jan Schacher. Motion to gesture to sound: mapping for interactive dance. In *Proceedings of the 2010 Conference on New Interfaces for Musical Expression*, pages 250–254, Sydney, 2010.
- [3] R. Laban (organized by L. Ullmann). *Domínio do Movimento*. Summus, 1978 (translated from the 1960 English version).
- [4] B. Ran et al. Multitask learning for Laban Movement Analysis. In *Proceedings of the 2nd International Workshop on Movement and Computing*, pages 37–44, Vancouver, 2015.
- [5] L. Maranan et al. Designing for movement evaluating computational models using LMA efforts qualities. In *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems*, pages 991–1000, New York, 2014.
- [6] L. Zhao. *Synthesis and acquisition of Laban Movement Analysis qualitative parameters for communicative gestures*. PhD thesis, University of Pennsylvania, 2012.
- [7] L. Zhao et al. Acquiring and validating motion qualities from live limb gestures. *Graphical Models*, 67:1–16, 2005.
- [8] R.N. Bindiganavale. *Building parameterized action representations from observation*. PhD thesis, University of Pennsylvania, 2000.
- [9] L. Naveda et al. “Topos” toolkit for pure data: exploring the spatial features of dance gestures for interactive musical applications. In *Proceedings of the Joint Conference ICMC-SMC*, pages 470–478, Athens, 2014.
- [10] S. Alaoui et al. Movement qualities as interaction modality. In *Proceedings of the Designing Interactive Systems Conference*, pages 761–769, New Castle, 2012.
- [11] I. Stravinsky. *Petrushka, for orchestra*. Dover, 1988 (1912 version).
- [12] I. Stravinsky. *Petrushka, for orchestra*. Cleveland Orchestra, conducted by G. Szell. Sony Classics, 1992.

# Impact of Algorithmic Composition on Player Immersion in Computer Games: A Case Study Using Markov Chains

Raul Paiva de Oliveira<sup>1\*</sup>, Tiago Fernandes Tavares<sup>1</sup>

<sup>1</sup>Faculdade de Engenharia Elétrica e Computação – Universidade Estadual de Campinas  
Cidade Universitária “Zeferino Vaz” – 13 083-970 Campinas, SP

raulpo@dca.feec.unicamp.br, tiagoft@dca.fee.unicamp.br

## Abstract

The feeling of immersion is an important aspect of gaming experiences. It can be greatly impacted by background music. In this work, we investigate the use of algorithmically-generated background music as a mean to generate immersion in gaming experiences. For such, we developed two versions of the same game. One of them uses music written by a human composer. The other uses real-time generated melodies based on a Markov chain. We evaluated immersion related to each of these versions using user questionnaires and performance measures. The results did show only a small immersion difference between the versions. This indicates that algorithmic music can be a suitable option for game content generation.

## 1. Introduction

Algorithmic music generation can be a method for fast content creation while preserving great variability. In this paper, we evaluate the impact of algorithmic music on the feeling of immersion in digital games and compare it with that related to human-composed music.

Immersion in digital games has been defined in many different ways. Brown and Cairns [1] defined immersion as one's degree of involvement with a game. Ermi and Mäyrä [2] state that immersion could be defined as a sensation in which one's perception is completely taken over, being fully enveloped by an alternative reality. Brockmyer et al. [3] also refers to immersion as the potential to induce feelings of being part of the game environment, and that most video game players report experiencing some degree of immersion.

\*Supported by FAEPEX.

Immersion is a phenomenon that can be linked to diverse objective measurements [4]. For this reason, we took into consideration various proposed methods to measure the player's awareness of their own immersion level. The experiment described in this work was based on the one described in Jenet's [5] and Zhang and Fu's [6] works.

In this experiment, the participants performed five activities: 1) solving a tangram puzzle, 2) playing the experimental game; 3) answering a questionnaire regarding immersion; 4) playing the game again, and 5) solving the same tangram puzzle as in step 1. Thus, the only difference between the control and test groups was the type of music playing during the game sessions (steps 2 and 4). Immersion levels were evaluated in both groups.

The results showed that the group of users playing the game with traditionally composed music had a slight higher immersion level than the group with procedurally generated music. However, the results didn't show a statistically significant difference in average immersion levels between the control and test group. This can mean that procedurally generated music, albeit a little less effectively, can be a viable alternative for game music production.

This work is organized as follows. Section 2 discusses related work. The experiments performed in this work, as well as their results, are described in Section 3. A deeper discussion is conducted in Section 4 and, finally, Section 5 concludes the paper.

## 2. Related Works

A great number of works related to immersion in digital games and algorithmic generated music

has been conducted in the last years. Jennet et al.[5] argues that players tend to have more difficult on switching from an immersive activity to a non-immersive one. Thus, after playing the game their performance on solving the tangram puzzle should be worse. To test this hypothesis the participants were asked to solve a tangram puzzle before and after a game session. Also, after the game session the players answered an immersion questionnaire. On the control group, instead of playing a game, the participants performed a computer task that involved clicking on squares appearing on the computer screen. The author's results indicates that, on average, the participants in the test group took more time to complete the tangram after playing the game, hence indicating that they had a higher immersion level.

Zhang and Fu [6] proposed an experiment to test if players would experience different levels of immersion playing a game with or without background music. Their experiment was based on Jennet's, comprising solving puzzles, playing the game for a pre-defined length and answering a questionnaire. The results showed that players in the group with background music reported experiencing a higher feeling of immersion.

Video games can be experiences with a very dynamical audiovisual environment. Thus, techniques for generating musical content procedurally could be exploited to deliver a more diversified material, or even one that bears a more meaningful relationship with the player's experience.

The GenJam model proposed by Biles [7] was inspired on the typical behavior of an improviser musician in the Jazz style. It uses genetic algorithms to generate phrases and melodies in real time, using a sequence of symbols representing musical notes in the form of discrete events. Combined, they form hierarchically related melody pieces, that is, measures, formed by musical notes, and phrases, formed by measures. In this model, the fitness of each individual (measure of phrase) is estimated by the behavior of the musician that interacts with the system.

Also, statistical models generated interesting results, as it is the case of the OMax project [8].

It uses a multi-level Markov chain to simulate improvisation. Thus, after receiving MIDI events from a musician, it generates musical pieces with similar characteristics.

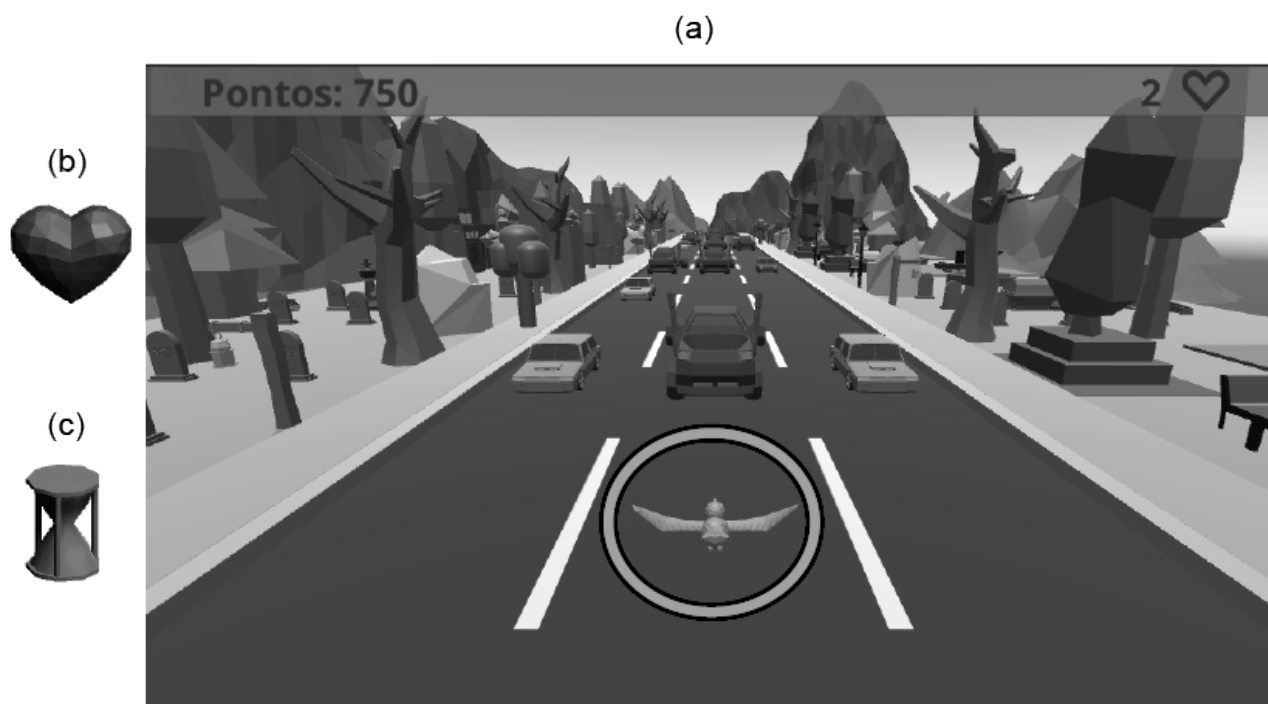
Livingstone and Brown [9] propose in his experiment an environment with adaptive musics in digital games, altering musical elements in real time to communicate the desired emotions. To achieve this, it employs a set of rules relating emotions to these musical elements. According to the author, it successfully communicates emotions such 'Happy', 'Sad' and 'Dreamy'.

Besides that, models proposed for computer-assisted composition, such as Mcvigar, Fukuyama and Goto [10], have showed promising results. This algorithm proposed by the authors analyzes rhythmic and melodic characteristics from transcriptions of musics played by guitarists from different styles. Then, these characteristics are employed to generate new musical pieces. Despite being different than the originals, these new musical elements have a great style similarity, being able to be used in a convincing way. Such work provides valuable insight towards the creation of procedural content in an interactive gaming environment.

### 3. Experiment

The game used in this work, as shown in Figure 1-a, is an infinite runner - that is, a game in which the character runs in an infinite track and must avoid obstacles and collect bonuses that helps their survival. In our game, the character loses one life point for each collision with a vehicle, and in turn recovers one life point by collecting a heart bonus (Figure 1-b). An hourglass bonus slows down vehicles to make it easier to avoid them (Figure 1-c).

Figure 2 shows the use of the two different background music types. For the Control Group, the pre-composed music in MIDI format was simply played by the game engine in an infinite loop. To generate the music for the Test Group, the same MIDI file used in the control group was split into two musical note lists representing its two melodies, and each list was further split into a list of note frequencies and a list



**Figure 1: Experimental Game Screenshot.** The character is inside the highlighted circle (a). The character has to avoid the vehicles and collect the bonus hearts (b) and hour-glasses (c).

of note durations. After that, we generated four Markov Chain models for the original music, one for each of the frequency or note lists. After this, each pair of models was used to generate a frequency and duration, joined into a musical note. The two new notes would then be played, and the note generation process would be repeated as needed. Figure 3 shows an example of a Markov Chain modeling these events. The pre-composed music was a two-instrument arrangement of the ‘Green Hill Zone’ stage music, from the game ‘Sonic the Hedgehog’. Figure 4-a shows an excerpt from the original music used for the control group. Figure 4-b to Figure 4-e show four example segments generated with the Markov Chain for the control group.

### 3.1. Participants

The sequence of steps proposed on this experiment was based on previous work by Jenet et al. [5] and Zhang and Fu [6]. In our experiments, we invited 38 participants students from FATEC Americana and divided them in two groups of 19. This research was approved by

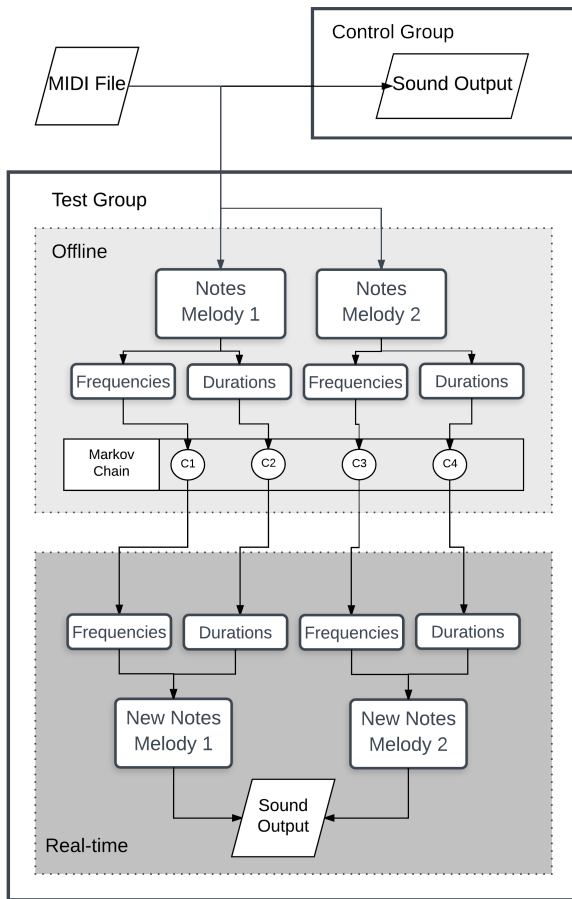
the Ethics Committee at the UNICAMP. All the participants were Technology of Game Development course students, with diverse age and gender.

### 3.2. Structure and Procedure

During the experiment, each participant was instructed to run the program to start the following procedures: 1) solving a tangram puzzle, 2) playing the game; 3) answering a questionnaire regarding immersion; 4) playing the game again, and 5) solving the same tangram puzzle as in step 1. The time limit for solving the tangram puzzle was defined as five minutes, due to time constraints related to the experiment location.

After 10 minutes of playing the game, the session was interrupted for answering the questionnaire. It contains 31 questions relating to immersion, with answers ranging from 1 (none) to 5 (very much).

After answering the whole questionnaire, the participant returns to the game session for another 10 minutes. Finally, after this session the player must solve the same tangram puzzle as in



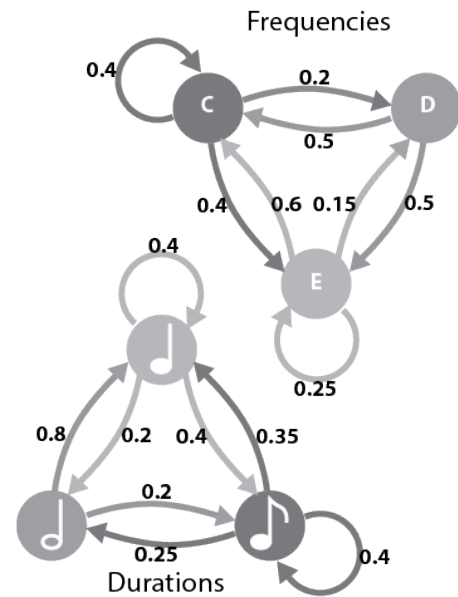
**Figure 2: Process used to generate background music for the control group (MIDI loop) and the test group (procedurally generated music).**

the first step. Figure 5 summarizes the experiment steps.

Performance data was also collected throughout the whole player's participation. This data comprises best and worst score in both game sessions, the time interval required to solve the first and second puzzles, and the performance improvement in solving the puzzle.

### 3.3. Results

We analyzed the interaction between elements that indicate a higher level of immersion with the different types of music to which the participants were exposed during game sessions using data collected on task execution and answers from the questionnaire. Figure 6 presents the means and standard deviations of each of the evaluated aspects in the control and test groups, and Figure 7



**Figure 3: Example Markov Chain that can be used to automatically generate content.**

shows the mean and standard deviations related to each question of the immersion questionnaire.

As shown in Figure 6, the participants in the control group (with pre-composed music) presented a greater mean score in their responses in the immersion questionnaire. However, the group with algorithmically generated music had a smaller gain in performance on solving the puzzle for the second time. This can indicate a greater cognitive effort while playing on the intermediate step, making it more difficult to fixate or transfer the knowledge acquired on the first attempt to solve the tangram puzzle.

## 4. Discussion

The idea that music generated by an algorithm could be linked to a higher cognitive effort is reinforced by the most favorable answers in the group with algorithmically generated music to questions such as 'Have you felt you were giving your best?'. It also suggests that this feeling is not exactly the same as the one accentuated by the pre-composed music. The latter would be more related to a feeling of emotional involvement, or of an experience with a stronger aesthetic component. This may be verified with

Figure 4 consists of five systems of musical notation, labeled (a) through (e). System (a) is a control melody excerpt, consisting of two staves (treble and bass clef) in 4/4 time. The tempo is marked as  $\text{♩} = 120$ . The melody is written in the treble clef, and the bass line is a simple eighth-note accompaniment. A 'Recor' label is visible in the top right of the first staff. Systems (b), (c), (d), and (e) are example melodies generated for the experiment group. Each system consists of two staves (treble and bass clef) in 4/4 time. The melodies are written in the treble clef, and the bass lines are simple accompaniments. The melodies are variations of the control melody, showing different rhythmic and melodic patterns.

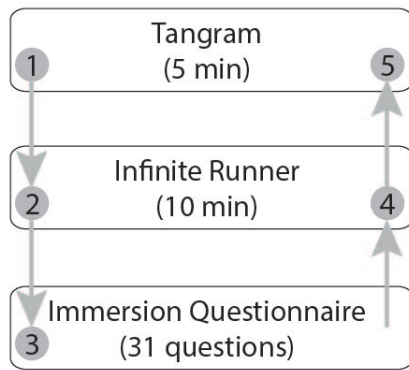
**Figure 4: Control melody excerpt (a) and four example melodies generated for the experiment group (b-e).**

the group's answers being more favorable to the questions 'How much you would say you enjoyed playing the game?' or 'To what extent did you feel emotionally attached to the game?'.

This result contradicts the findings of the ex-

periments made by [5] and [6], in which a higher score in the immersion questionnaire was related to a larger increase in performance when solving the puzzle. This suggests that the presence of music generated by algorithms would be tied





**Figure 5: Flowchart describing the experiment. It begins with a tangram puzzle, proceeds to game playing, then to a immersion questionnaire. After that, game playing is repeated and the tangram puzzle is solved again.**

to a higher cognitive effort while gaming, but not necessarily to the totality of the more broad phenomenon described as immersion.

It is also worth noting that the overall immersion score mean was not high. That fact could be attributed to the total length of the experiment (from 30 to 50 minutes). This could lead to users feeling fatigued, and thus affecting their responses. Also other factors during the experiment execution could have impacted the results, such as noises and other distractions from other participants.

## 5. Conclusion

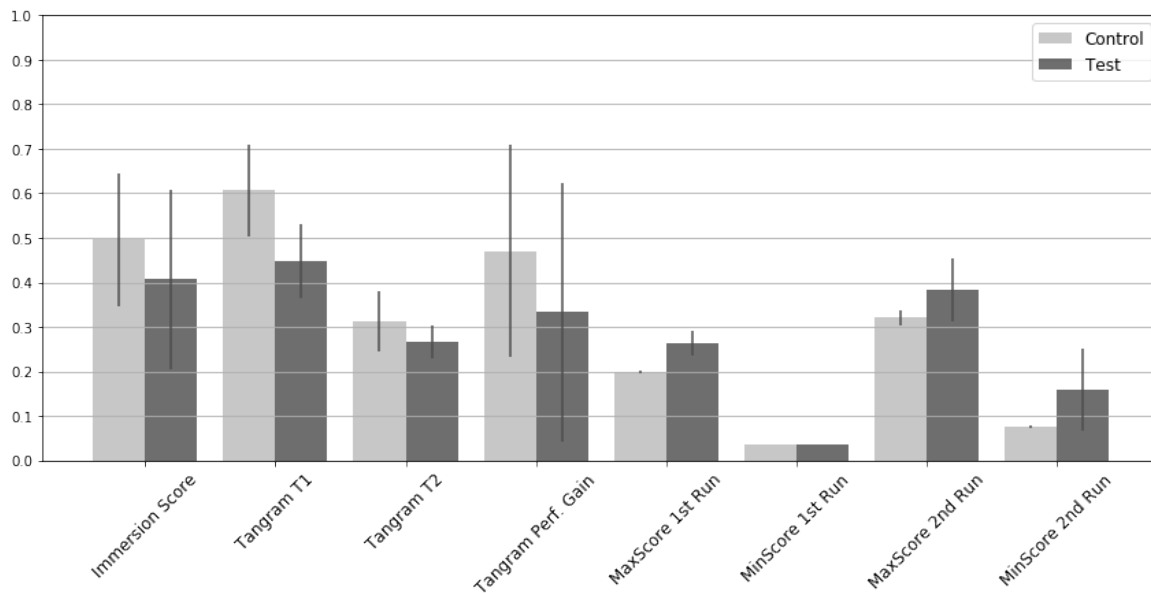
In this work, we conducted an experiment to evaluate the impact of using algorithmically-generated background music in player immersion in digital games. For such, we performed a user study in which half of the subjects played a game with human-composed music and the other half played a similar game, with computer-generated music. Results show that algorithmic music reinforces cognitive aspects of immersion, whereas human-composed music is linked to higher aesthetic and emotional aspects of immersion. This suggests that algorithmically composed music can be a viable alternative to looping music for game content generation.

Furthermore, this work raises questions about the impact of algorithmically generated music in game immersion. We speculated that different genres of games could be affected differently. Also, different compositional techniques could be employed to observe the effects from procedurally generated music with an emphasis on the harmonic structure, or even the instrument or timbre choices. Investigation on these topics will be conducted in future work.

## References

- [1] Emily Brown and Paul Cairns. A grounded investigation of game immersion. In *CHI'04 extended abstracts on Human factors in computing systems*, pages 1297–1300. ACM, 2004.
- [2] Laura Ermi and Frans Mäyrä. Fundamental components of the gameplay experience: Analysing immersion. *Worlds in play: International perspectives on digital games research*, 37:2, 2005.
- [3] Jeanne H Brockmyer, Christine M Fox, Kathleen A Curtiss, Evan McBroom, Kimberly M Burkhart, and Jacquelyn N Pidruzny. The development of the game engagement questionnaire: A measure of engagement in video game-playing. *Journal of Experimental Social Psychology*, 45(4):624–634, 2009.
- [4] Raul Paiva de Oliveira, Denise Calsavara Paiva de Oliveira, and Tiago Fernandes Tavares. Measurement methods for phenomena associated with immersion, engagement, flow, and presence in digital games. SBGames, 2016.
- [5] Charlene Jennett, Anna L Cox, Paul Cairns, Samira Dhoparee, Andrew Epps, Tim Tijs, and Alison Walton. Measuring and defining the experience of immersion in games. *International journal of human-computer studies*, 66(9):641–661, 2008.
- [6] Jiulin Zhang and Xiaoqing Fu. The influence of background music of video games on immersion. *Journal of Psychology & Psychotherapy*, 2015, 2015.
- [7] John a. Biles. GenJam: A genetic algorithm for generating jazz solos. *Proceedings of*





**Figure 6: Mean and standard deviation by parameter and by test or control group**

*the International Computer Music Conference*, pages 131–137, 1994.

- [8] G Assayag, G Bloch, and M. Chemillier. Omax-Ofon. *Sound and Music Computing*, 2006.
- [9] Steven R Livingstone and Andrew R Brown. Dynamic response: real-time adaptation for music emotion. In *Proceedings of the second Australasian conference on Interactive entertainment*, pages 105–111. Creativity & Cognition Studios Press, 2005.
- [10] Matt Mcvicar, Satoru Fukuyama, and Masataka Goto. AutoRhythmGuitar : Computer-aided Composition for Rhythm Guitar in the Tab Space. (September):293–300, 2014.

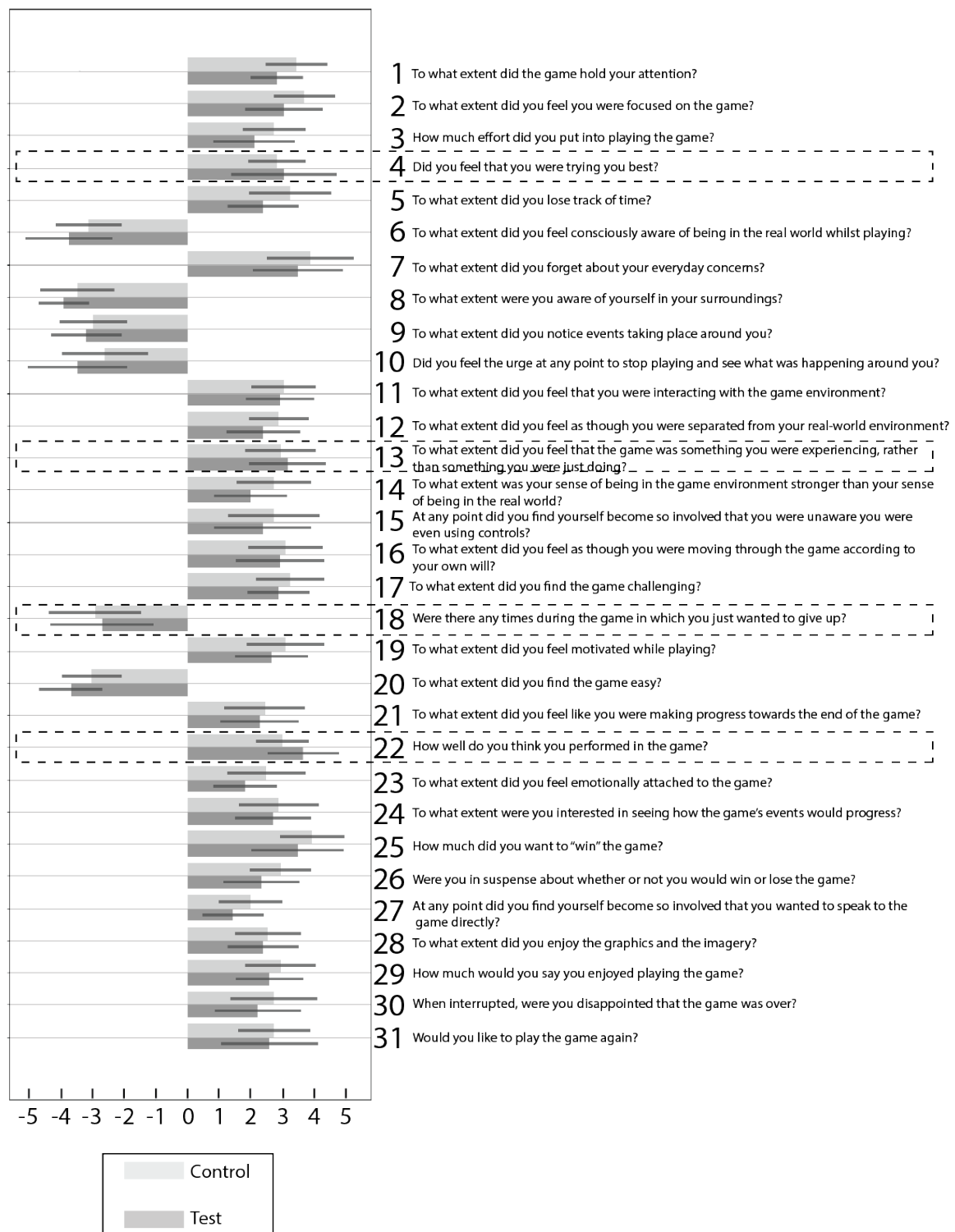


Figure 7: Mean and standard deviation for answers to each question in the immersion questionnaire.

# Impact of Genre in the Prediction of Perceived Emotions in Music

Felipe Souza Tanios<sup>12\*</sup>, Tiago Fernandes Tavares<sup>12†</sup>

<sup>1</sup>Intedisciplinary Nucleus for Sound Studies

University of Campinas

R. da Reitoria, 165, Cidade Universitária – 13083-590 – Campinas - SP

<sup>2</sup>Laboratory of Digital Signal Processing for Telecommunications

School of Electrical and Computer Engineering

University of Campinas Av. Albert Einstein, 400 – 13083-970 – Campinas - SP

first\_ftanios@nics.unicamp.br, second\_tavares@dca.fee.unicamp.br

## Abstract

In on-line music streaming systems and music digital libraries, the emotions that people perceive while listening to a track have become an important criterion while trying to find what to listen to. There are several possible solutions that use emotion as a criterion. Some of them separate and predict emotions in a track learning individually from its user which demands a considerable amount of time to generate adequate result. Others predict in a generalized way for every user, so it does not take into account musical preferences specifically of its user. People that identify with a social group tend to perceive the same emotions in music and social groups' members often identify with the musical genre. In this paper we describe a method for music emotions prediction using genre information and compare it with a similar classifier that does not use genre information. Results show that prediction accuracy improves in all tested genres, except for one. This suggests that music in different genres convey emotions using different means.

## 1. Introduction

Online music streaming systems provide a large audio dataset to users. An effective search method is important to deal with such a large amount of data. One possible criterion for music search is the emotion users perceive in music [1].

This work analyzes music as a socio-cultural construct. In this context, music can be seen as a

method for communication and social interaction [2]. Furthermore, music genres are often part of the identity of a group that shares common interests [3].

People that belong to the same social group are surrounded by the same social construct. Since emotions are socio-cultural constructs [4], we can reckon that people in the same social group share the similar emotion perceptions.

We use Hanslick's approach regarding music and emotions [5]. He argues that music itself does not bring an embedded emotion. Instead, it carries a "musical idea", which evokes a perceived emotion on its listener. Perceived emotions are more likely to be used in this type of study because the assessment of "felt" emotion requires personal and situational factors to be considered [6].

There are several proposed models for the description of emotions. The categorical model [7] of Ekman's "Big Six"[8] describes six transculturally perceived emotions (surprise, fear, disgust, anger, happiness and sadness). However, experiments conducted by Livingstone and Brown [9] show that music can consistently communicate happy, sad and dreamy emotions, but this consistency was not found in other emotions.

The automatic prediction of perceived emotion has often relied on mapping pieces to a content-inspired vector space in which close vectors depict similar tracks. This approach was proposed for automatic music genre classification [10] and then adapted to the prediction of other labels, such as emotions [1].

In our work, we evaluated two different struc-

\*Supported by PIBIC-UNICAMP.

†Supported by FAPESP

tures for automatic prediction of perceived emotion. In the first, we simply applied audio texture-based classification [10] using perceived emotion as labels. In the second one, we trained a different classifier for each genre in the dataset. This experiment is further discussed in Section 2. The results are shown in Section 3. Finally, Section 4 concludes the paper.

## 2. Method

The classifier implemented in this work is based on Tzanetakis and Cook’s algorithm for music genre classification [10]. The classifier extracts framewise features (spectral centroid, spectral roll-off, spectral flatness, spectral flux, and 30 MFCCs) from a track and calculates their mean and average over a 1s-long sliding window. After that, it calculates the mean and average of the means and averages for each feature within each window. This yields a 136-dimensional vector that maps the track to a content-meaningful  $\mathbb{R}^{136}$  vector space. Thus, tracks that are near to each other in this vector space are more similar to each other in terms of auditory texture content. After mapping, the algorithm performs supervised classification within this vector space. In our work, we used the well-known Support Vector Machine (SVM) algorithm for classification.

Our tests aimed at evaluating the impact of using genre information on the prediction of perceived emotion in music. For this purpose, we developed a dataset containing audio files and their respective genre and perceived emotion labels.

To build our dataset we used anonymous discussion websites with graded comments (e.g. Reddit<sup>1</sup>), so we could fetch the average group’s emotion perceived for each track. We gathered tracks that were more highly voted in the discussion threads. The first dataset consisted of 200 tracks divided into three emotions, namely Happy, Sad and Dreamy, following previous work by Livingstone and Brown [9].

While analyzing the dataset it became clear to us that the community usually agreed the classi-

fication happy and sad tracks but did not agree about dreamy tracks. This left us with a total of only 17 dreamy tracks. Because this is much smaller number than the 90 sad tracks and 93 happy tracks, we decided not to use this emotion in our dataset.

The dataset consist of 183 full-length tracks, divided into five different genres (Indie-Rock, Jazz, Heavy-Metal, Bossa-Nova and Classical music), labeled according to two different perceived emotions. The number of tracks for each genre and emotion label is shown in Table 1.

Genre	Number of tracks		
	Sad	Happy	Total
Jazz	18	19	37
Heavy-Metal	18	19	37
Indie-Rock	18	19	37
Classical	18	17	35
Bossa Nova	18	19	37

**Table 1: Number of tracks for each genre and emotion label in the dataset.**

We performed two different tests, as shown in Figure 1. The first one uses the whole dataset to train the same classifier, so it would not consider genre information for emotion prediction. The second one trains a different classifier for each genre. All tests were performed using a 10-fold cross validation schema.

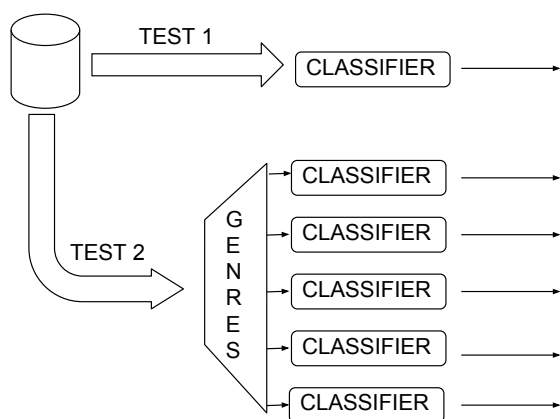
The results of both tests are further discussed in the next section.

## 3. Results and Discussion

The average F1-Score results for Test 1 and Test 2, as defined in Figure 1, are shown in Table 2. As it can be seen, the F1-score in Jazz, Heavy-Metal, Indie-Rock and Classical Music are better than the result related to not taking genre into account. The result for Bossa-Nova, however, is worse than all the others.

We performed a T-Test comparing the results related to each genre-specific classification process and the results related to the classification using all genres at the same time. The result

<sup>1</sup><http://www.reddit.com/>



**Figure 1: Experiment Flowchart.** In test 1, we use the dataset containing all genres to train a unique classifier. In test 2, we train a different classifier for each genre.

Test	Genre	F1-score	P
1	All	$0.77 \pm 0.10$	-
2	Jazz	$0.85 \pm 0.21$	0.26
	Heavy-Metal	$0.79 \pm 0.14$	0.11
	Indie-Rock	$0.85 \pm 0.24$	0.76
	Classical	$0.86 \pm 0.24$	0.25
	Bossa Nova	$0.56 \pm 0.21$	0.0005

**Table 2: Mean and standard deviations of F1-scores of each genre individually and all genres together. The reported P-Values are related to a t-test between the results related to the classifiers for each genre and the classifier using all genres.**

of this test, reported in Table 2, shows that only Bossa-Nova had a statistically significant difference ( $P < 0.05$ ) related to the full set.

This result can be related to the fact that all used features are related to timbre, thus lyrics are ignored. Classical music is commonly built on timbre variations, using instrumentation variation and harmony techniques as compositional tools. Likewise, Heavy-Metal, Indie-Rock and Jazz are musical styles in which frequently there are perceptible timbre differences between happy and sad songs. Bossa-Nova, on the other hand, is a genre in which acoustic guitar and voice are frequently used instruments and the rhythm and intonations are more uniform throughout the

style[11]. This means that, although listeners can perceive different emotions in Bossa-Nova song, they cannot be adequately predicted using timbre information.

Therefore, our results indicate that emotion prediction systems should rely on genre information in order to achieve better results. This corroborates with the mixed texture and lyrics methodology used by Yang and Chen [12] and the ideas used as basis by Hu and Downie [1]. However, these results show that improvements on emotion prediction due to the use of both lyrics and texture are related to the genre specificities of each dataset. Also, these results suggest that lyrics and textures should be weighted differently in the prediction of perceived emotion in each genre.

The next section presents conclusive remarks.

#### 4. Conclusion

In this work, we evaluated the impact of using genre information in the prediction of perceived emotion in music. For such, we created a dataset in which tracks were labeled according to their genre and their perceived emotion. Our results show that using a specific classifier for each genre yields better results than using a single classifier, with no genre information, for all genres, except for Bossa-Nova.

This means that musical texture is not an effective feature to explain the perceived emotion in Bossa-Nova. Thus, emotions in different genres are perceived through different means. Therefore, genre information can be used to improve automatic prediction of perceived emotions, which can potentially improve the accuracy of online music suggestion systems. This poses an interesting direction for future work.

#### References

- [1] Xiao Hu and J. Stephen Downie. Improving mood classification in music digital libraries by combining lyrics and audio. In *Proceedings of the 10th Annual Joint Conference on Digital Libraries, JCDL*

- '10, pages 159–168, New York, NY, USA, 2010. ACM.
- [2] Yi-Hsuan Yang and Homer H. Chen. Prediction of the distribution of preceived music emotions using discrete samples. *IEEE Transactions on Audio, Speech and Language Processing*, 19(5):2184–2196, 2002.
- [3] Beatriz Ilari. Música, comportamento social e relações interpessoais. *Psicologia em Estudo*, 11(1):191–198, 4 2006.
- [4] MAURO GUILHERME PINHEIRO KOURY. *Emoções, sociedade e cultura*. EDITORA CRV, 2009.
- [5] Eduard Hanslick. *Do Belo Musical*. Editora UNICAMP, 1989.
- [6] Alf Gabrielsson. Emotion perceived and emotion felt: Same or different? *Musicae Scientiae*, 5(1\_suppl):123–147, 2001.
- [7] Paula Dornhofer Paro Costa. *Two-Dimensional Expressive Speech Animation - Animação 2D de Fala Expressiva*. PhD thesis, Universidade Estadual de Campinas - Faculdade de Engenharia Elétrica e de Computação, 2 2015.
- [8] P. Ekman and W. Friesen. *Facial Action Coding System: A Technique for the Measurement of Facial Movement*. Consulting Psychologists Press, Palo Alto, 1978.
- [9] Steven R. Livingstone and Andrew R. Brown. Dynamic response: Real-time adaptation for music emotion. pages 105–111, 2005.
- [10] George Tzanetakis and Perry Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, 2002.
- [11] Bryan McCann. Blues and samba: Another side of bossa nova history. *Luso-Brazilian Review*, 44(2):21–49, 2007.
- [12] Dan Yang and WonSook Lee. Disambiguating music emotion using software agents. In *Proceedings of the 5th International Conference on Music Information Retrieval*, Barcelona, Spain, October 10-14 2004.

# Sounderfeit: Cloning a Physical Model with Conditional Adversarial Autoencoders

Stephen Sinclair<sup>1</sup>

<sup>1</sup>Inria Chile

Av. Apoquindo 2827, piso 12 Las Condas – Santiago, Chile

`stephen.sinclair@inria.cl`

## Abstract

An adversarial autoencoder conditioned on known parameters of a physical modeling bowed string synthesizer is evaluated for use in parameter estimation and resynthesis tasks. Latent dimensions are provided to capture variance not explained by the conditional parameters. Results are compared with and without the adversarial training, and a system capable of “copying” a given parameter-signal bidirectional relationship is examined. A real-time synthesis system built on a generative, conditioned and regularized neural network is presented, allowing to construct engaging sound synthesizers based purely on recorded data.

## Introduction

This paper explores the use of an *autoencoder* to mimic the bidirectional parameter-data relationship of an audio synthesizer, effectively “cloning” its operation.

The *autoencoder* is an artificial neural network (ANN) configuration in which the network weights are trained to minimize the difference between input and output, in essence learning the identity function. When forced through a bottleneck layer of few parameters, the network is made to represent the data with a low-dimensional “code,” which we call the latent parameters.

Recently adversarial configurations have been proposed as a method of regularizing this latent parameter space in order to match it to a given distribution [1]. The advantages are two-fold: to ensure the available range is uniformly covered, making it a useful interpolation space; and to maximally reduce correlation between parameters, encouraging them to represent orthogonal

aspects of the variance. For example, in a face-generator model, this could translate to parameters for hair style and the presence of glasses [2].

Meanwhile, it has also been shown that a generative network can be conditioned on known parameters [3], to make it possible to control the output, for example, to generate a known digit class when trained on MNIST digits.

In this work, these two concepts are combined to explore whether an adversarial autoencoder can be conditioned on known parameters for use in both parameter estimation and resynthesis tasks. In essence, we seek to have the network simultaneously learn to mimic the transfer function from parameters to data of a periodic signal, as well as from data to parameters, using adversarial training to regularize the distribution of the latent space. Latent dimensions are provided to the network to capture variance not explained by the conditional parameters, usually referred to in the image synthesis literature as “style”; in audio, they may represent internal state, stochastic sources of variance, or unrepresented parameters e.g. low-frequency oscillators.

Results are visualized and some preliminary evaluations are performed. Most problems came from issues with the dataset rather than with the network architecture or training algorithms, and we conclude with some lessons learned in the art of “synth cloning” and how to handle sampling. A real-time synthesis system, Sounderfeit, built on a generative, regularized neural network is presented, allowing to construct engaging sound synthesizers based purely on recorded data, with optional conditioning on prior knowledge of parameters.

## Configuration

### Dataset

Given a network with sufficient capacity we can encode any functional relationship, but for the experiments described herein a periodic signal specified by a small number of parameters was sought that nonetheless features some complexity and is related to sound synthesis. Thus, a physical modeling synthesizer proved a good choice. We used the bowed string model from the *STK Synthesis Toolkit in C++* [4], which uses digital waveguide synthesis and is controlled by 4 parameters: *bow pressure*, the force of the bow on the string; *bow velocity*, the velocity of the bow across the string; *bow position*, the distance of the string-bow intersection from the bridge; and *frequency*, which controls the length of the delay lines and thus the tuning of the instrument.

The parameters are represented in STK as values from 0 to 128, and thus we do not worry about physical units in this paper; all parameters were linearly scaled to a range  $[-1, 1]$  for input to the neural network. The data was similarly scaled for input, and a linear descaling of the output is performed for the diagrams in this paper. Additionally, the per-element mean and standard deviations across the entire dataset were subtracted and divided respectively in order to ensure similar variance for each discrete step of the waveform period.

To extract the data, a program was written to evaluate the bowed string model at 48000 kHz for 1 sec for each combination of *bow position* and *bow pressure* for integers 0 to 128. The *bow velocity* and *volume* parameters were both held at a value of 100. For each instance, the last two periods of oscillation were kept, and since some parameter combinations did not give rise to stable oscillation, recordings with an RMS output lower than  $10^{-5}$  (in normalized units) over this span were rejected, giving a total of 15731 recordings evenly distributed over the parameter range. The frequency was selected at 476.5 Hz to count 201 samples to capture two periods—some parameter combinations changed the tuning slightly, but inspection by eye of 50 periods concatenated end to end showed minimal deviation at this frequency for a wide variety of parameters. Two

periods were recorded in order to minimize the impact of any possible reproduction artifacts at the edges of the recording during overlap-add synthesis. The recordings were phase-aligned using a cross-correlation analysis with a representative random sample, then differentiated by first-order difference, and 200 sample-to-sample differences were thus used as the training data, normalized as stated above. Reproduction thus consists of de-normalizing, concatenating (using overlap-add with a Blackman window) and first-order integrating the final signal. This dataset we refer to as *bowed1*.

Recently-published similar work recommends using log-spectrum representations rather than raw audio [5], however we found that since we are concentrating on small two-period, phase-aligned samples featuring relatively small amount of variance (as compared to trying to learn a large dataset of fully mixed instruments), learning the raw audio signal was no problem. In future work it is possible that different/better results could be had by using a log-spectrum representation, but in this manner we avoided the need to perform phase reconstruction. The use of a differentiated representation also helped to suppress noise.

As will be discussed below, parameter estimation on new data was not successful based on this dataset. To resolve this, a second extended dataset, *bowed2*, was created in a similar manner, however instead of recording only the steady state portion, the synthesizer was executed continuously while changing the parameters randomly at random intervals. This allowed to capture more dynamic regimes. 100,000 samples uniformly covering the parameter range were captured for *bowed2*.

### Learned conditional autoencoding

While the principle job of the autoencoder is to reproduce the input as exactly as possible, in this work we also wish to estimate the parameters used to generate the data. Thus we additionally *condition* part of the latent space by adding a loss related to the parameter reconstruction. This is somewhat different to providing conditional parameters to the *input* of the encoder [1, 3].



Note that the presence of the latent parameters is what allows for the fact that we do not assume that the signal is purely deterministic in the known parameters. For instance, in a physical signal there may be internal state variables that are not taken into account in the initial conditions, or acoustic characteristics such as room reverb that are not considered a priori. Naturally, the less deterministic the signal is in the known parameters, the more must be left to latent parameters, and the poorer a job we can expect the parameter reconstruction to do.

### Generative adversarial regularization

The code used in the middle layer of an autoencoder, called the *latent parameters*, which we shall refer to as  $z$ , when trained to encode the data distribution  $p(x)$ , has conditional posterior probability distribution  $q(z|x)$ . As mentioned, it is in general useful to regularize  $q(z|x)$  to match a desired distribution.

Several methods exist for this purpose: a *variational autoencoder* (VAE) uses the Kullback-Leibler divergence from a given prior distribution. Other measures of difference from a prior are possible. The use of an adversarial configuration has been proposed [1] to regularize  $q(z)$  based on the negative log likelihood from a discriminator on  $z$ .

With adversarial regularization, a discriminator is used to judge whether a posterior distribution  $q(z)$  was likely produced by the generator and is thus sampled from  $q(z|x)$ , or rather sampled from an example distribution  $p(z)$ , which is often set to a normal or uniform distribution. The discriminator is itself an ANN which outputs a 1 for “real” samples of  $p(z)$  or a 0 for “fake” samples of  $p(z) = q(z|x)$ . The training loss of the generator, which is also the encoder of the autoencoder, maximizes the probability of fooling the discriminator into thinking it is a real sample of  $p(z)$ , while the discriminator simultaneously tries to increase its accuracy at distinguishing samples from  $p(z)$  and samples from  $q(z|x)$ . Thus the encoder eventually generates posterior  $q(z|x)$  to be similar to  $p(z)$ .

### System Architecture Summary

Putting together the above concepts, the system is composed of two neural networks and three training steps.

First, the autoencoder network is composed of the encoder  $E = f(x)$  and the decoder/generator  $G = g(z, y)$ . The discriminator is designed analogously as  $D = h(z)$ . For notational convenience, we also define  $G_E = g(E) = g(f(\mathbf{x}))$ ,  $D_E = h(E_z)$ , and  $D_z = h(\mathbf{z})$  where  $\mathbf{x} = x_1 \dots x_s$  are sampled from  $p(x)$ ,  $\mathbf{z} = z_1 \dots z_s$  is sampled from  $p(z)$ , and  $s$  is the batch size.  $E_z(x)$  and  $E_y(x)$  are the first  $n$  and the last  $m$  dimensions of  $f(x) \in [z_1 \dots z_n \ y_1 \dots y_m]$ , respectively. In the current work,  $f(x)$  and  $g(z, y)$  are simple one-hidden-layer ANNs with one non-linearity  $\zeta$  and linear outputs:

$$f(x) = \zeta(x \cdot w_1 + b_1) \cdot w_2 + b_2 \quad (1)$$

$$g(z) = \zeta(z \cdot w_3 + b_3) \cdot w_4 + b_4 \quad (2)$$

$$h(z) = \zeta(z \cdot w_5 + b_5) \cdot w_6 + b_6 \quad (3)$$

We used the rectified linear unit  $\zeta(x) = \max(0, x)$ , but had similar results with tanh nonlinearities.

The data, described below, was composed of 200-wide 1-D vectors, and we had acceptable results using hidden layers of half that size, so  $w_1 \in \mathbb{R}^{200 \times 100}$ ,  $w_2 \in \mathbb{R}^{100 \times (n+m)}$  and  $w_3, w_5 \in \mathbb{R}^{(n+m) \times 100}$ ,  $w_4, w_6 \in \mathbb{R}^{100 \times 1}$ , where  $(n+m)$  was 2 or 3, depending on the experiment. The bias vectors  $b_1 \dots b_6$  had corresponding sizes accordingly. Hyperparameter random search was used to guide, but not automatically select hyperparameters.

The training steps were performed in the following order for each batch:

1. A stochastic gradient descent (SGD) optimiser with a learning rate of 0.005 was used to train the full set of autoencoder weights  $w_1 \dots w_4$ , and  $b_1 \dots b_4$ , minimizing both the data  $x$  reconstruction loss and parameter  $y$  reconstruction loss,  $\mathcal{L}_{AE}$  by back-propagation. The weighting parameter  $\lambda = 0.5$  is described below.
2. SGD with learning rate 0.05 was used to train the generator weights and biases

$w_1$ ,  $w_2$ ,  $b_1$ , and  $b_2$ . The negative log-likelihood  $\mathcal{L}_G$  was minimized by back-propagation.

3. SGD with learning rate 0.05 was used to train the discriminator weights and biases  $w_5$ ,  $w_6$ ,  $b_5$ , and  $b_6$ . The negative log-likelihood  $\mathcal{L}_D$  was minimized by back-propagation.

where,

$$\mathcal{L}_{AE} = \sum (\mathbf{x} - g(f(\mathbf{x})))^2 + \lambda \sum (\mathbf{y} - g(\mathbf{x}))^2 \quad (4)$$

$$\mathcal{L}_G = - \sum \log(D_E) \quad (5)$$

$$\mathcal{L}_D = - \sum (\log(D_z) + \log(1 - D_E)). \quad (6)$$

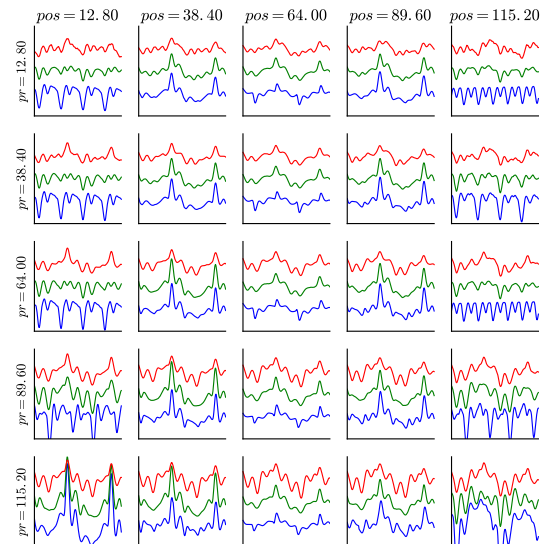
Experiments were performed using the TensorFlow framework [6], which implemented the differentiation and gradient descent (back-propagation) algorithms. A small batch size of 50 was used, with each experiment evaluated after 4,000 batches. It was found that smaller batch sizes worked better for the adversarial configuration, since the updates of each step are interleaved. Matrices  $\mathbf{z}$  and  $\mathbf{x}, \mathbf{y}$  were sampled independently from  $\mathbf{Z} \sim p(\mathbf{z}) = \mathcal{U}(-1, 1)$  and  $(\mathbf{X}, \mathbf{Y}) \sim p(x, y)$  for each step, where  $\mathcal{U}(a, b)$  is the uniform distribution in range  $[a, b]$  inclusive.

## Experiments

Six conditions were tested in order to explore the role of conditional and latent parameters. The number of known parameters in the dataset was 2. We tried training the *bowed1* dataset with and without an extra latent parameter. We label these conditions  $D1_Z2_Y$  and  $D0_Z2_Y$  respectively. The third condition,  $N1_Z2_Y$ , was like the  $D1_Z2_Y$  condition but without adversarial regularization on  $q(z|x)$ .

In order to further understand how latent parameters may help capture unknown variance, we tested a condition of treating *bow pressure* as a known parameter and leaving *bow position* to be captured by  $z$ . This was accomplished simply by repeating  $D1_Z2_Y$  with one missing  $y$  parameter, thus labeled  $D1_Z1_Y$ .

Finally, to compare conditioning with the “natural” distribution of the data among latent



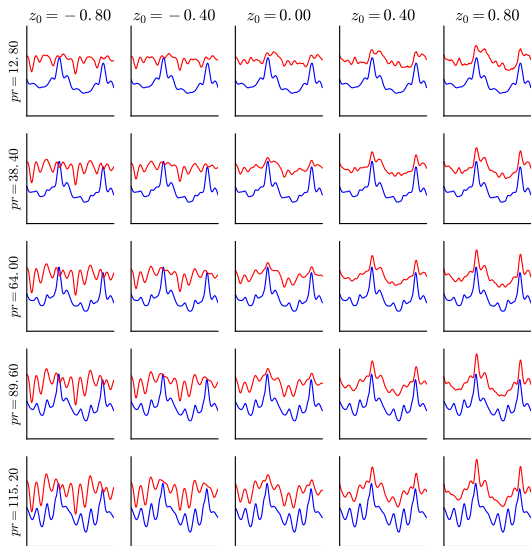
**Figure 1: Output of  $D1_Z2_Y$  as *pressure*  $y_0$  and *position*  $y_1$  are changed, with  $z_0 = 0$ . Top (red) is the decoder, middle (green) is the autoencoder, bottom (blue) is the dataset.**

parameters and the effects of adversarial regularization thereupon, two configurations with no conditional parameters, with and without the discriminator, were explored, named  $D2_Z0_Y$  and  $N2_Z0_Y$  respectively.

## Results

Figure 1 demonstrates the results of  $D1_Z2_Y$ . Comparing the middle and bottom curves, we can see that while it has some trouble with low values of *bow pressure* and the extremes of *bow position*, the autoencoder is able to more or less encode the distribution in our dataset. The top curve (red) was generated by explicitly specifying the same parameters instead of letting the autoencoder infer them, and demonstrates the output for parameter-driven reconstruction if  $z_0$  is held constant. Although not a perfect reproduction, this demonstrates that parameters were conditioned according to the dataset, and thus the ANN models the data-parameter relationship.

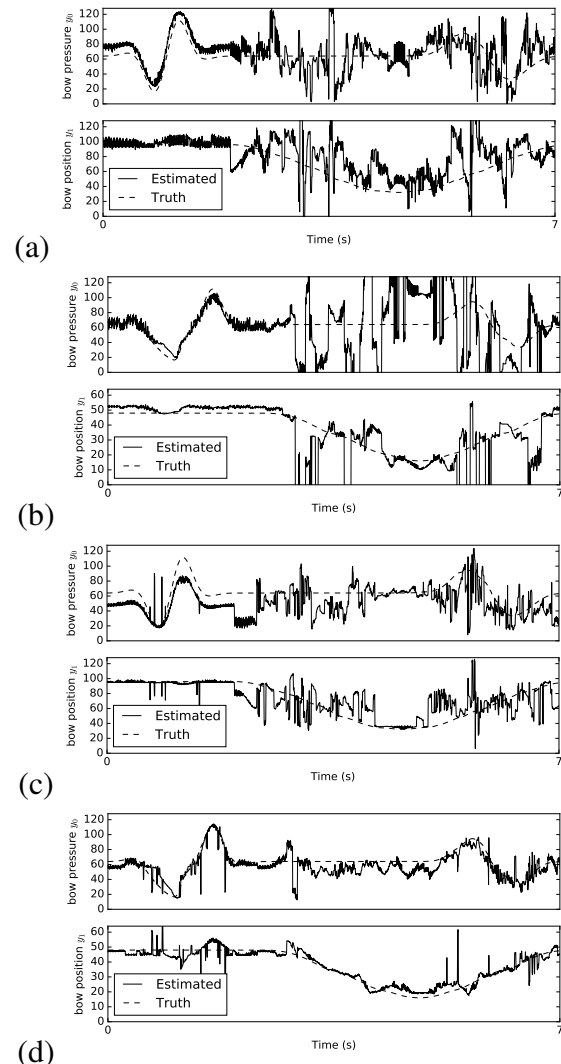
The role of  $z$  is now considered in Figure 2, by holding *bow position* constant ( $y_1 = 100$ ) and examining how the signal changes with  $z_0$ .



**Figure 2: Output of  $D1_{z2Y}$  as bow position is set to 100, and bow pressure and latent  $z_0$  variable are changed. Top (red): Decoder output; bottom (blue): dataset.**

One notices that for some values of  $z_0$  the signal matches well, and for others it varies from the target signal. For example, we can see that in this case, high values of  $z_0$  push the signal towards two sharp peaks, while low values of  $z_0$  tend towards more oscillations; both  $z_0 = -0.8$  and  $z_0 = 0.8$  resemble the  $pr = 115.2$  condition, but in different aspects. Meanwhile there is consistency with the “stylistic” influence of  $z_0$  on the signal for different values of bow pressure.

Finally, we look at the encoder (parameter estimator) performance, by producing a *new* signal from the synthesizer with a parameter trajectory starting with variation only in *bow pressure* and then variation only in *bow position*, and then variation in both parameters. Figure 3(a) shows actually rather disappointing performance in this respect, however it does clarify some information not present in the previous analysis: the estimation is clearly better for *bow pressure*, but easily disturbed by changes in *bow position*. Nonetheless we see the *tendency* of the estimate in the right direction, with rather a lot of flipping above and below the center. Since varying the hyperparameters of our network did not



**Figure 3: Parameter estimation performance of the  $D1_{z2Y}$  network for (a) *bowed1* full dataset, RMS error=23.23; (b) *bowed1* half dataset, RMS error=33.45; (c) *bowed2* full dataset, RMS error=19.54; (d) *bowed2* half dataset, RMS error=8.61.**

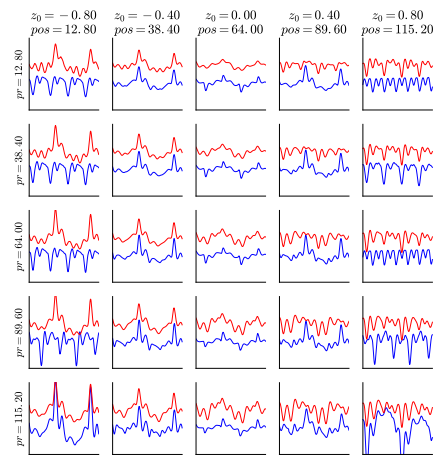
solve this problem, we hypothesized that this error could come from two sources: (1) ambiguities in the dataset—indeed, if one examines the shape of the signal as *bow position* changes, one notices a symmetry between values on either side of  $pos=64$ . By consequence the inverse problem is underspecified, leading to ambiguity in the parameter estimate. (2) underrepresented variance in the dataset; the new testing data varies continuously in the parameters, but the dataset was constructed based on the per-parameter steady state.

To investigate this, the network was trained on a “half dataset”, consisting only of samples of *bowed1* where *bow position*  $< 64$ . Furthermore, as mentioned, an extended dataset, *bowed2*, was constructed based on random parameter variations.

Results in Figure 3(b)-(d) show that training on the half-*bowed1* dataset changed the character of errors, but did not improve overall, however the extended *bowed2* dataset gave improved parameter estimation, and much improved in the *bow position*  $< 64$  case. Thus it can be concluded that both sources contributed to parameter estimation difficulties.

We also examine using  $D1_Z1_Y$  how the network performs if only 1 parameter is conditioned. In Figure 4 it can be seen that the signals match in many cases, very similar indeed to Figure 2. However, this is not a given, since the parameter on the horizontal axis, *bow position*, was not conditioned! Indeed, it is reflected by the  $z_0$  variable automatically, since it is the principle source of variance unexplained by the conditioned parameter *bow pressure*.

Figure 5 shows the resulting parameter space if both parameters are left to be absorbed by the unsupervised latent space. Indeed it seems that with regularization, the system is encouraged to cover the entire range of variance in the dataset. Without regularization ( $N2_Z0_Y$ ) we see some relationship between the two inferred variables  $z_0$  and  $z_1$  (Fig. 6)—although it appears more complex than could be captured by a Pearson’s correlation—while this is completely gone for the regularized version ( $D2_Z0_Y$ ). The star shape is generated because without regulariza-



**Figure 4: Top (red): Output of  $D1_Z1_Y$  as *bow pressure*  $y_0$  and the latent variable  $z_0$  are changed; bottom (blue):  $pr$  and  $pos$  from the dataset.**

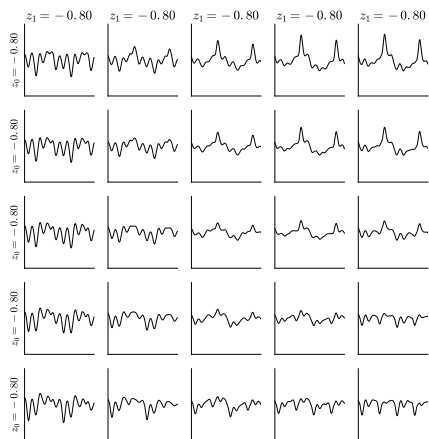
tion, the autoencoder attempts to maximally separate various aspects of the variance in a reduced 2-dimensional space, which can be useful for data analysis but does not produce a good interpolation space.

Finally, we found that with this small decoder network of  $100 \times 3$  weights and 100 biases, an overlap-add synthesis could be performed in real time on a laptop computer (10 seconds took 8.5 seconds to generate), and we can thus create a data-driven wavetable synthesizer, which we call *Souderfeit*, with a number of adjustable parameters. The regularization encourages the parameter space to be “interesting,” in the sense that they represent orthogonal axes within the distribution that cover a defined range and tend towards uniform coverage without “holes.” This is demonstrated in Figure 7.

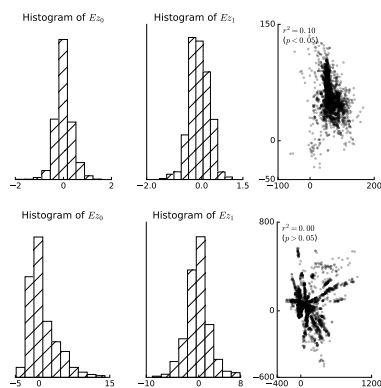
## Conclusions

These experiments showed some modest success in copying the parameter-data relationship of a physical modeling synthesizer.

Like many machine learning approaches, the quality of results depend strongly on the hyperparameters used: network size and architecture, learning rates, conditional regularization weights, etc., and these must be adapted to the

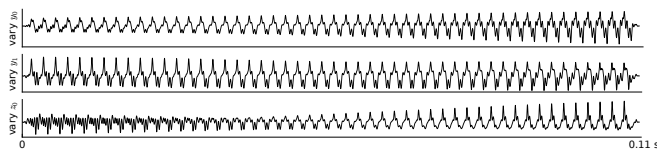


**Figure 5: Output of  $D2z0y$ , varying  $z_0$  and  $z_1$ .**



**Figure 6: Parameter distributions: top,  $D2z0y$  (regularized); bottom,  $N2z0y$  (no regularization).**

dataset. Shown are results from the best parameters found after some combination of automatic and manual optimisation on this specific dataset, which we use to demonstrate some principles of the design, however it should be noted that actual results varied sometimes unexpectedly with small changes to these parameters. This hyperparameter optimization is non-trivial, especially when it comes to audio where mean squared error may not reveal much about the perceptual quality of the results, and so a lot of trial and error is the game. Thus, a truly “universal”, turn-key synthesizer copier would require future work on measuring a combined hypercost that balances well the desire for good reproduction with good parameter estimation quality, and well-distributed latent parameters. Such work could go beyond mean squared error to involve



**Figure 7: Overlap-add output of  $D1z2y$ , varying each parameter over a short interval.**

perceptual models of sound perception.

Some practical notes: (1) We found that getting the adversarial method to properly regularize the latent variables in the presence of conditional variables is somewhat tricky; the batch size and relative learning rates played a lot in balancing the generator and discriminator performances. New research in adversarial methods is a current area of investigation in the ML community and many new techniques could apply here; moreover comparison with variational methods is needed. (2) Expectedly, we found the parameter estimation extremely sensitive to phase alignment; we tried randomizing phase of examples during training, which gave better parameter estimates, but this was quite damaging to the autoencoder performance. In general oversensitivity to phase is a problem with this method, a downside to the time domain representation.

Nevertheless we have attempted to outline some potential for use of autoencoders and their latent spaces for audio analysis and synthesis based on a specific signal source. Only a very simple fully-connected single-layer architecture was explored; myriad improvements could likely be made using convolutional layers, different activation functions, etc. More important than the quality of these specific results, we wish to point out the modular approach that autoencoders enable in modeling oscillator periods of known and unknown parameters, and that, in contrast to larger datasets covering many instruments [5], interesting insights can be had even on small data.

The motivation of the work could be questioned, in the sense that a black box model seemingly does not bring much to the table in the presence of an existing, semantically-rich physical model. Indeed, in this work a digital synthesizer was used as an easy way to gain access to a

fairly complicated but clean signal with a small number of parameters. In principle this method could be used on much richer, real instrument recordings, provided that a measurement or estimate of acoustically-relevant parameters is available. For example, we have applied it to recorded vowel vocalizations, with the vowel category as a single continuous variable, creating a real-time vowel synthesizer similar to the bowed string results with a controllable vowel knob and other characteristics represented by the latent space.

Another question may be why perform simultaneous estimation and generation with the same network. Indeed, part of the long-term goals are to “play” somewhat more with the latent space, such as using it for what is known in the audio community as cross-synthesis, or in the machine learning community as “style transfer”, i.e., swapping the bottom and top halves of two such autoencoder networks, allowing to drive a synthesizer by both conditioned and latent parameters estimated on an incoming signal.

## References

- [1] A. Makhzani et al. Adversarial autoencoders. In *Proc. Int. Conf. Learning Representations*, 2016.
- [2] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *preprint arXiv:1511.06434*, 2015.
- [3] J. Gauthier. Conditional generative adversarial nets for convolutional face generation. Class project for Stanford CS231N: Convolutional neural networks for visual recognition, winter semester, 2014.
- [4] Perry R. Cook and Gary P. Scavone. The Synthesis ToolKit (STK). In *Proc. Int. Comp. Music Conference*, October 1999.
- [5] J. Engel et al. Neural audio synthesis of musical notes with WaveNet autoencoders. *preprint arXiv:1704.01279*, 2017.
- [6] M. Abadi et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Available: <http://tensorflow.org>.

# Synesthesia Add-on: a Tool for HTML Sonification

Roberto Piassi Passos Bodo<sup>1\*</sup>, Flávio Luiz Schiavoni<sup>2</sup>

<sup>1</sup> Institute of Mathematics and Statistics (IME)  
University of São Paulo (USP)  
São Paulo, São Paulo, Brazil

<sup>2</sup>Department of Computer Science (DCOMP)  
Federal University of São João Del Rei (UFSJ)  
São João Del Rei, Minas Gerais, Brazil

rppbodo@ime.usp.br, fls@ufsj.edu.br

## Abstract

Web browsers using HTML5 and WebAudio have been widely used as real-time audio environment and brought up new possibilities for web art. In this paper, we present an investigation on HTML sonification. In our approach, HTML pages are read as musical scores and page elements are played as a sequencer. We developed a tool for website sonification which can be used to explore musical creativity and to create new sound contexts based on the web pages. We present the tags and attributes that are mapped to sound parameters. In addition, we show how was created sounds and visual feedback in this tool.

## 1. Introduction

For long, Internet has been used to distribute art pieces through virtual Museums and virtual Art Galleries. Although these are a great relationship between art and this technology, there are more possibilities to art on the web than being a place to art commerce, release or distribution.

From the very first moment that images went out over the Web, artists have been using the web as an art medium and not just a new way to publish information. It can be used with artistic purpose in a response to the digitalization of cultural forms and the information technology revolution [1] [2]. This art instance, called **web art**, is about art works made specifically for the web, available all the time, every place and everywhere to several distinct participants [2].

\*Supported by CAPES.

Historically, Web Art is probably a continuation of Media Art where the viewer is not only part of the audience, but a participant in that experience.

An important aspect on Web Art is about the processes and tools used to this art instance. Web Art is relatively inexpensive to produce because HTML is a free language, HTTP is a free protocol and web browsers are free tools. It makes this art format very accessible to digital artists[1].

Since the web browser is the web art medium, a basic construction on Web Art usually runs over the Hyper Text Markup Language (HTML). HTML is a fast evolving language, supported by a considerable community of developers, which evolution demanded the incorporation of new tags and routines in its last version, namely HTML5. Apart from several news, HTML5 brought to the web a sound engine, called WebAudio API[3]. Before HTML5 it was already possible to play audio files in a web page. The WebAudio API brought the possibility to create and process real-time audio in the web browser. The browser is now a real-time audio rendering tool and it gave new possibilities of interaction and feedback to Web Art.

In this paper, we are using this APT to extend the browser capability to artistically sonify every HTML page using the WebAudio API. Classically, a goal of sonification is to transform complex multidimensional data in intuitive audio [4] [5] (as in text-to-speech or web accessibility tools).

Alternatively to this approach for sonification, our goal here is to create purely aesthetic sounds



that can be used to inspire compositional process or just be enjoyed by the user.

The remainder of the paper is organized as follows: Section 2 presents the idea of Web pages sonification, Section 3 presents the project main implementation, Section 4 presents initial results, and Section 5 presents Conclusion and Future Works.

## 2. About Web Pages

In the beginning of the Internet era, a web page was a plain text, black on white, unformatted and very informative document. In fact, there are several old websites which the content is presented in this format even today. Despite the fact that websites changed a lot from this initial layout, this format keeps the content easily rendered by a text-to-speech converter or a Braille device (and accessibility is the most common goal in web sonification [6]).

Such tools will read only the text content and will ignore the visual aspects of the website, namely a set of invisible tags and attributes whose settings will define the page style.

Nowadays, we consider a traditional web page as a triple of HTML, CSS, and JavaScript resources. Basically, HTML is responsible for the page content, page structure, CSS for styling (positions, sizes, colors, etc), while JavaScript is used for dynamic actions in client side (interactions, animations, etc).

In our project, we are ignoring the page content. We approached a web page as a music sheet and, with that in mind, the first idea we had was to use page structure as music structure and page styling as some kind of flavor for synthesis.

## 3. Implementation

Since the genesis of this project, we wish to allow users to sonify any site on the web. Naturally, host our code together with all the sites around the web is not possible, so we have decided to extend the browser adding to it a sonification capability.

Browsers can be extensible by an add-on, a kind of plugin written in JavaScript, and users

can install add-ons adding more functionalities to the browser.

To ensure cross-browser compatibility we choose the WebExtensions system <sup>1</sup> which is compatible with Google Chrome and Opera natively, and with Firefox and Microsoft Edge after a few modifications.

With the add-on we could run our JavaScript code in any website, allowing users to explore sonification ideas around the web. The new add-on added our functionality to the web browser, as presented on Figure 1.

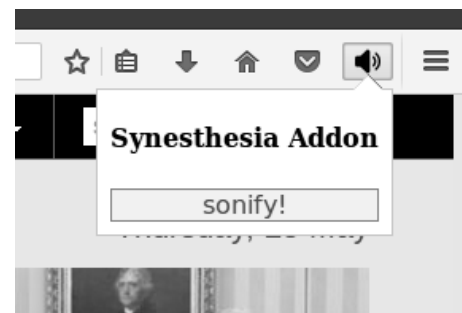


Figure 1: Screenshot of the add-on

### 3.1. Collecting information about websites

The HTML language has the peculiarity of enabling two sites with completely different code structures to have the same visual, and two sites with very similar source code to have completely different visual. Furthermore, it is not easy to determine which HTML tags are used in a website just by observing the rendered page.

To have a better view about which tags and attributes are common and how we could sonify it, we decide to first collect information about sites. We analyzed the HTML structure of a few websites generating statistics of all the information we were interested in.

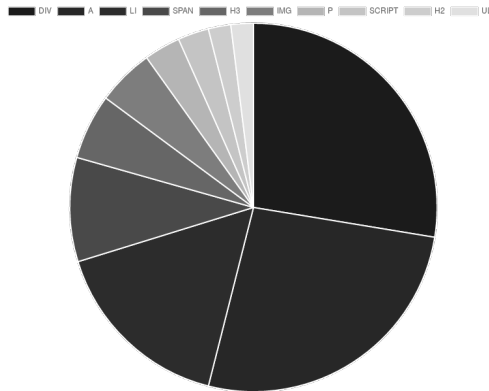
More specifically, we made a JavaScript code that goes down recursively on the page structure from the `<body>` element gathering all useful information and saving it in an array (making a linearization of the data).

For instance, this code calculates a histogram of tag occurrences. We selected the top 10 tags

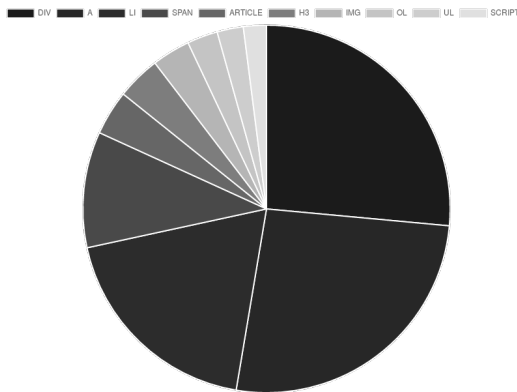
<sup>1</sup>Available on <https://developer.mozilla.org/en-US/Add-ons/WebExtensions>



for <http://bbc.com> and <http://cnn.com>, and presented them on Figure 2 and 3.



**Figure 2: Most used tags of BBC site and the proportion of their occurrences.**



**Figure 3: Most used tags of CNN site and the proportion of their occurrences.**

According to these statistics, the most used tags in these websites are DIV, A, LI, and SPAN (other tags appears more rarely). It gave us a good tip about which HTML tags we could use in the sonification process.

When an element is selected to be sonified, we only have its name and where it belongs on the HTML tree, and we need more information about it to map to the synthesizers parameters. So, we decided to investigate all of the style attributes that are associated with the HTML element.

Analyzing the CSS sources is an arduous task, because we can have more than one class by element, more than one file defining these classes, and rule overwriting due to inheritance (the latter is the worst case).

So we decided to work with computed style in DOM. For this, there is a JavaScript function called *getComputedStyle()* that returns the values assigned to hundreds of style attributes. This large number was, to us, a new challenge due to the need to select a subset of all of this information. We considerate the so-called box model to solve this problem.

Every modern browser has in its developer tools a graphic visualization of an element by the terms of the box model. It considers that every element behave like a box. When you inspect an element, this graphic shows the computed dimensions (width and height) beside other attributes like padding (internal gap), margin (external gap) and border. Some browsers also show the top and left positions of the element.

With this in hand and with the hypothesis that these are the most used attributes in a page, we move on to the step of fetching elements from real sites and taking statistics from their styles. Tables 1 and 2 present the statistics of the same two websites mentioned above.

**Table 1: BBC global statistics of styles**

attribute	avg	std	min	max
width	343.83	820.57	0	25000
height	138.57	420.53	0	7589
top	1407.89	4013.39	-100000	7563
left	124.13	3014.81	-100000	1600
padding	19.38	56.80	0	702
borderWidth	0.11	0.84	0	18
margin	3.88	13.46	-224	88

**Table 2: CNN global statistics of styles**

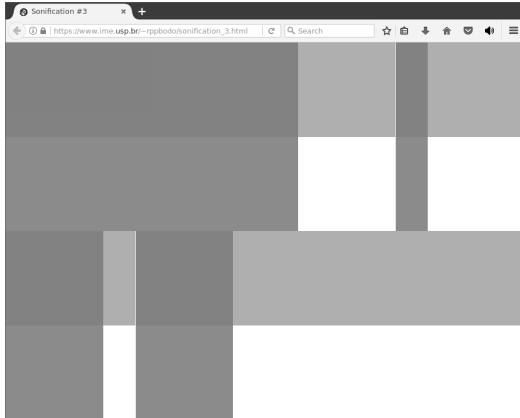
attribute	avg	std	min	max
width	198.98	300.08	0	1348
height	53.89	120.86	0	1899
top	543.99	1051.76	-10000	2844
left	147.54	289.62	-100	1348
padding	4.97	29.78	0	639
borderWidth	0.12	0.75	0	8
margin	-0.66	30.05	-626	100

### 3.2. The HTML Score

HTML is a hierarchical document where tags are organized in a defined order. We can look for a complete HTML structure as a tree with nodes and their children, having the `<body>` tag as the

root. In our sonification project, the order of the elements defines the order of notes reproduction and the attributes of the elements define the notes configurations.

For instance, the pitch of the note can be defined by one attribute; the duration by other; the amplitude by yet another (specific information about mapping can be found in subsection 3.3).



**Figure 4: Example page where each block is mapped to a note in our sonification add-on.**

What is important here is that was created a voice for each tag. Let's imagine a synthesizer playing notes for each rectangle on the web page shown in Figure 4 (implemented only with divs). When we map width to pitch and height to duration, we will have notes such as those in Figure 5. If we had a more complex page with more tags, we calculate the most frequent ones and sonify each tag using a different synthesizer.



**Figure 5: Visualization of notes in a piano roll after the sonification of the page presented in Figure 4**

Observe that the notes in the piano roll do not have rests between them. This happened because when the time of a note offset is reached, the next one was started immediately. But this behavior is not mandatory. We can map any attribute to

notes' starting times and with this we can produce rests between notes. This will work because the starting time of a note, added with its duration, not necessarily will reach the starting time of the next one.

Even more, with this onset time mapping, we can produce polyphony also. Depending of which attribute is mapped to onset time, we can have several notes beginning at the same time. The reader is already able to perceive the numerous behaviors that the add-on will have according to all possible mappings.

The notes' scheduling was implemented using a JavaScript method called *setTimeout()* that receives a callback function and a time value. With this method we were able to schedule all of the notes onsets. The offsets were handled by the synthesizers themselves (we passed the duration to *startNote()* procedure).

This scheduling is totally dependent of a parameter that we called "virtual clock relative speed". The concept of virtual clock was presented by Roger Dannenberg in 1984 [7] in the context of Automated Musical Accompaniment. Basically, it is a clock that evolves by a rate of the real clock. This rate is defined by the parameter mentioned above, which is a floating point number that will determine the speed of the execution. For instance, if it is set to 0.5, the speed it will be half of the original, if it is set to 2, it will be the double, and so on.

The most exciting thing about this parameter is not the capability of changing speed, but the capability of changing speed by attribute mapping. We could map body's background color to the speed and have darker pages slower than brighter ones. Or we could have variable speed throughout the sonification. This can be achieved mapping one attribute from the current sounding element to dynamically change speed for the next one. This can lead to totally unanticipated behavior, but this is considered a positive feature of the add-on for us.

### 3.3. Mapping HTML attributes to sound information

Initially, we did not know which attribute would be mapped to which synthesizer parame-

ter. So, using the extracted statistics, we detected the minimum and maximum values for each attribute and mapped the actual range of values to [0, 1].

With all the values between 0 and 1, we could map any of them to any synthesis parameter using any linear function (to adjust the value ranges). In our first tests we realized that some style attributes are most commonly used than others, even when we work with the set of the box model attributes.

For instance, borders are not frequently used (lots of elements had `borderWidth` equals to zero). Paddings and margins are the next ones in number of zero occurrences, followed by top and left. The most frequent computed style from our selection are, definitely, width and height.

When we use width and height in a parameter we had a greater variety; when we use `borderWidth` in this same parameter we had lots of equal behaviors. So we had two options: combine parameters (and, thus, have the less variable one working as a harmless modulator) or simply avoid setting these less frequent attributes to core synthesizers parameters (such as, pitch or duration).

### 3.4. Building Synthesizers

For the audio to be synthesized in real-time on the browser itself, we used `WebAudio` to implement the synthesizers. We tried to select techniques that cover various different timbres. Six instruments were developed for our add-on: Block, Drum, Guitar, Harpsichord, Maraca and a Pong.

All of them were implemented with the same interface so that they could be easily alternated. The pitch parameter is a MIDI note number, the duration is in milliseconds and amplitude is a value between 0 and 1 that determines the maximum in the note envelope. For illustration, the current mappings are presented on Table 3.

### 3.5. Visual Feedback

In our first implementation we realized that there were no visual feedback from the add-on. Because of it, it was almost impossible to precisely affirm which element was being sonified

**Table 3: Current properties mapping to sound attributes.**

Synth	pitch	duration	amplitude
1	$32 + W * 32$	$500 + H * 500$	T
2	$32 + L * 32$	$500 + P * 500$	T
3	$32 + H * 32$	$500 + W * 500$	L
4	$65 + L * 65$	$1000 * P * 5$	M
5	$32 + L * 32$	$500 + P * 500$	B
6	$44 + W * 32$	$500 + H * 500$	L

W = width / H = height / T = top / L = left

P = padding / M = margin / B = `borderWidth`

at each time. To solve it, a CSS class was created to visually highlight the elements that were emitting sounds at a moment.

First, we tried to change colors, but some elements did not accept that (the concept of background color doesn't match with an image tag, for instance). After that, we tried to add a very noticeable border, but that broke the layout.

Borders has width and some pixels were added in the overall width of the elements. With this, some side-by-side elements became misaligned or, even worse, did not fit at all on their original positions.

The solution was to work with 3D transformations that did not change the original layout and can be applied in any element. We implemented an animation that shakes the element using small translations from side-to-side, making one element quite prominent in the page.

With this visual feedback we found another issue: elements present in the HTML but invisible on the page were being sonified. For instance, the page can have an image carousel that has several images but only one is visible or it can have a dropdown menu that has its items hidden. We found several instances of this problem and we found very difficult to come up with a solution suitable to all cases.

Just to soften this issue, we implemented a heuristic to check if an element is visible. It has 3 steps: a) calculate the top-left and bottom-right corners of the element; b) find out what are the visible elements of the page on these points; c) check if both are the same original element.

We know that this heuristic is not perfect, be-

cause it does not solve, for instance, the case of a  $N, M$  object on top of a  $N + 1, M + 1$  object with 1px offset (we consider that the one below can not be seen), but it solves most of the rough cases.

With all of these visual issues surpassed, we ran the sonification add-on on the same websites that we extracted the statistics, and we were really satisfied with the audible results.

## 4. Results

Our first result is a statistics add-on, which can generate all statistics mentioned in Section 3.1 to any website, and help musicians and sonifiers to collect information about web pages. This add-on generates images for some statistics and for the piano roll, and can generate  $\text{\LaTeX}$  tabular code also. This add-on is available on <https://github.com/rppbodo/statistics-addon>.

The second and main result is the sonification add-on itself, a sequencer that plays HTML pages as musical scores and allow users to listen page structure. It has 6 instruments in JavaScript / WebAudio with a common interface. These instruments uses different synthesis techniques like FM, AM, and additive. The add-on source code is available on <https://github.com/rppbodo/synesthesia-addon>.

## 5. Conclusion

In this paper, we presented a novel form to create music based on HTML pages by the means of a sonification add-on. For this, we have chosen to use the structure of HTML websites, instead of the content, to control production through synthesizers.

We presented our strategies to sonification, the statistics used to create the strategy and several considerations about our development process. Certainly, our strategies included subjective choices and the process counted on a previous experience to create synthesizers and choose good parameters mapping the page elements to sound.

As future work, we intend to explore websites musically and research how it can collaborate to music creativity on composition and improvisation. In the present implementation we are consider only the page structure but it could be interesting to sonify text nodes too. We are considering to create a third tool to achieve text sonification.

### 5.1. Acknowledgment

We would like to thank the Computer Music Research Group<sup>2</sup> and the Sonology Research Center (NuSom)<sup>3</sup> at the University of São Paulo, and the support from CAPES<sup>4</sup>.

## References

- [1] Mark Tribe, Reena Jana, and Uta Grosenick. *New media art*. Taschen London and Cologne, 2006.
- [2] Annette Weintraub. Art on the web, the web as art. *Commun. ACM*, 40(10):97–102, October 1997.
- [3] Chris Rogers. W3c webaudio api. <https://www.w3.org/TR/webaudio/>, 2015. Accessed: 2017-07-01.
- [4] Oded Ben-Tal and Jonathan Berger. Creative aspects of sonification. *Leonardo*, 37(3):229–233, 2004.
- [5] Thomas Hermann, Andy Hunt, and John G Neuhoff. *The sonification handbook*. Logos Verlag Berlin, 2011.
- [6] Lori Stefano Petrucci, Eric Harth, Patrick Roth, André Assimacopoulos, and Thierry Pun. Websound: a generic web sonification tool, and its application to an auditory web browser for blind and visually impaired users. *Proceedings of ICAD 2000*, pages 6–9, 2000.
- [7] Roger B. Dannenberg. An on-line algorithm for real-time accompaniment. In *ICMC*, pages 193–198. Michigan Publishing, 1984.

<sup>2</sup><http://compmus.ime.usp.br/>

<sup>3</sup><http://www.eca.usp.br/nusom/>

<sup>4</sup><http://www.capes.gov.br/>

# Technology Enhanced Learning of Expressive Music Performance

Rafael Ramirez<sup>1\*</sup>, Fabio Ortega<sup>1</sup>, Sergio Giraldo<sup>1</sup>

<sup>1</sup>Music and Machine Learning Lab  
Music Technology Group  
Universitat Pompeu Fabra  
Roc Boronat 138  
08018 Barcelona, Spain

{rafael.ramirez, fabiojose.muneratti, sergio.giraldo}@upf.edu

## Abstract

Learning to play music is mostly based on the master-apprentice model in which modern technologies are rarely employed and students' interaction and socialisation is often restricted to short and punctual contact with the teacher. This often makes musical learning a lonely experience, resulting in high abandonment rates. In the context of TELMI, an international project, which aims to address these issues by providing new multi-modal interaction paradigms for music learning and to develop assistive, self-learning, real-time feedback, complementary to traditional teaching, this paper focuses on the computational modelling of expressive music performance as a tool for music learning. We record a professional violinist and apply machine learning techniques to induce an expressive model the recordings. We use this model to generate feedback on expressive aspects of arbitrary pieces to violin students.

## 1. Introduction

Music education requires a long learning trajectory and intensive practice. Learning to play music is mostly based on the master-apprentice model in which the teacher mainly gives verbal feedback on the performance of the student. In such a learning model, modern technologies are rarely employed and almost never go beyond audio and video recording. In addition, the student's interaction and socialisation is often restricted to short and punctual contact with the

teacher followed by long periods of self-study, which often makes musical learning a lonely experience, resulting in high abandonment rates [1].

One of the most challenging skills that students must learn during their learning process is the ability to play expressively. This is usually learnt by imitating expert performers to gradually develop the own playing style. This is normally a long process where there is no general rules on how to go about it. Furthermore expressive instructions often vary considerably from teacher to teacher. TELMI is an international effort to address the challenges music instrument education poses. Concretely, TELMI aims to design and implement new multi-modal interaction paradigms and prototypes for music learning and training based on state-of-the-art audio processing and motion capture technologies, and to create a publicly available reference database of multimodal recordings with data analytics. This database, no matter how extended it is, will inevitably be non-exhaustive in the sense that it will fail to contain all possible pieces that any student may want to practice. The work described in this paper aims at extrapolating the recordings in the database by providing general expressive computational models able to generate feedback about expressive issues in arbitrary music pieces. The multi-modal recordings in the database are extended by applying machine learning techniques using database recordings as training data. We use sound analysis techniques based on spectral models [15] for extracting high-level symbolic features from the recordings. In particular, for characterising the performances used in this work, we are interested in inter-note features representing informa-

\*This work has been partly sponsored by the Spanish TIN project TIMUL (TIN2013-48152-C2-2-R), the European Union Horizon 2020 research and innovation programme under grant agreement No. 688269 (TELMi project), and the Spanish Ministry of Economy and Competitiveness under the Maria de Maeztu Units of Excellence Programme (MDM-2015-0502).

tion about the music context in which expressive events occur. Once the relevant high-level information is extracted we apply machine learning techniques [9] to automatically discover regularities and expressive patterns for each performer. We use these regularities and patterns in order to identify a particular performer in a given audio recording.

## 2. Background

Understanding and formalizing expressive music performance is an extremely challenging problem which in the past has been studied from different perspectives, e.g. [14], [4], [2]. The main approaches to empirically studying expressive performance have been based on statistical analysis (e.g. [12]), mathematical modeling (e.g. [17]), and analysis-by-synthesis (e.g. [3]). In all these approaches, it is a person who is responsible for devising a theory or mathematical model which captures different aspects of musical expressive performance. The theory or model is later tested on real performance data in order to determine its accuracy. The majority of the research on expressive music performance has focused on the performance of musical material for which notation (i.e. a score) is available, thus providing unambiguous performance goals. Expressive performance studies have also been very much focused on (classical) piano performance in which pitch and timing measurements are simplified.

Previous research addressing expressive music performance using machine learning techniques has included a number of approaches. Lopez de Mantaras et al. [6] report on SaxEx, a performance system capable of generating expressive solo saxophone performances in Jazz. One limitation of their system is that it is incapable of explaining the predictions it makes and it is unable to handle melody alterations, e.g. ornamentations.

Ramirez et al. [11] have explored and compared diverse machine learning methods for obtaining expressive music performance models for Jazz saxophone that are capable of both generating expressive performances and explaining the

expressive transformations they produce. They propose an expressive performance system based on inductive logic programming which induces a set of first order logic rules that capture expressive transformation both at an inter-note level (e.g. note duration, loudness) and at an intra-note level (e.g. note attack, sustain). Based on the theory generated by the set of rules, they implemented a melody synthesis component which generates expressive monophonic output (MIDI or audio) from inexpressive melody MIDI descriptions. With the exception of the work by Lopez de Mantaras et al and Ramirez et al, most of the research in expressive performance using machine learning techniques has focused on classical piano music where often the tempo of the performed pieces is not constant. The works focused on classical piano have focused on global tempo and loudness transformations while we are interested in note-level tempo and loudness transformations.

## 3. Melodic description

First of all, we perform a spectral analysis of a portion of sound, called analysis frame, whose size is a parameter of the algorithm. This spectral analysis consists of multiplying the audio frame with an appropriate analysis window and performing a Discrete Fourier Transform (DFT) to obtain its spectrum. In this case, we use a frame width of 46 ms, an overlap factor of 50%, and a Keiser-Bessel 25dB window. Then, we compute a set of low-level descriptors for each spectrum: energy and an estimation of the fundamental frequency. From these low-level descriptors we perform a note segmentation procedure. Once the note boundaries are known, the note descriptors are computed from the low-level values. The main low-level descriptors used to characterise note-level expressive performance are instantaneous energy and fundamental frequency.

Energy computation. The energy descriptor is computed on the spectral domain, using the values of the amplitude spectrum at each analysis frame. In addition, energy is computed in different frequency bands as defined in [5], and these values are used by the algorithm for note segmentation.

Fundamental frequency estimation. For the estimation of the instantaneous fundamental frequency we use a harmonic matching model derived from the Two-Way Mismatch procedure (TWM) [7]. For each fundamental frequency candidate, mismatches between the harmonics generated and the measured partials frequencies are averaged over a fixed subset of the available partials. A weighting scheme is used to make the procedure robust to the presence of noise or absence of certain partials in the spectral data. The solution presented in [7] employs two mismatch error calculations. The first one is based on the frequency difference between each partial in the measured sequence and its nearest neighbour in the predicted sequence. The second is based on the mismatch between each harmonic in the predicted sequence and its nearest partial neighbour in the measured sequence. This two-way mismatch helps to avoid octave errors by applying a penalty for partials that are present in the measured data but are not predicted, and also for partials whose presence is predicted but which do not actually appear in the measured sequence. The TWM mismatch procedure has also the benefit that the effect of any spurious components or partial missing from the measurement can be counteracted by the presence of uncorrupted partials in the same frame.

Note segmentation is performed using a set of frame descriptors, which are energy computation in different frequency bands and fundamental frequency. Energy onsets are first detected following a band-wise algorithm that uses some psycho-acoustical knowledge [5]. In a second step, fundamental frequency transitions are also detected. Finally, both results are merged to find the note boundaries (onset and offset information).

Note descriptors. We compute note descriptors using the note boundaries and the low-level descriptors values. The low-level descriptors associated to a note segment are computed by averaging the frame values within this note segment. Pitch histograms have been used to compute the pitch note and the fundamental frequency that represents each note segment, as found in [8]. This is done to avoid taking into account mis-

taken frames in the fundamental frequency mean computation.

Musical Analysis. It is widely recognized that humans perform music considering a number of abstract musical structures. In order to provide an abstract structure for the recordings under study, we decided to use Narmour's theory of perception and cognition of melodies [10] to analyze the performances.

## 4. Expressive Performance Modeling

### 4.1. Training Data

In this work we are focused on Celtic jigs, fast tunes but slower than reels, that usually consist of eighth notes in a ternary time signature, with strong accents at each beat. The training data used in our experimental investigations are monophonic recordings of nine Celtic jigs performed by a professional violinist. Apart from the tempo (he played following a metronome), the musicians were not given any particular instructions on how to perform the pieces.

### 4.2. Note Features

The note features represent both properties of the note itself and aspects of the musical context in which the note appears. Information about the note includes note pitch and note duration, while information about its melodic context includes the relative pitch and duration of the neighboring notes (i.e. previous and following notes) as well as the Narmour structures to which the note belongs. The note's Narmour structures are computed by performing the musical analysis described before. Thus, each performed note is characterized by the tuple

(Pitch, Dur, PrevPitch, PrevDur, NextPitch, NextDur, Nar1, Nar2, Nar3)

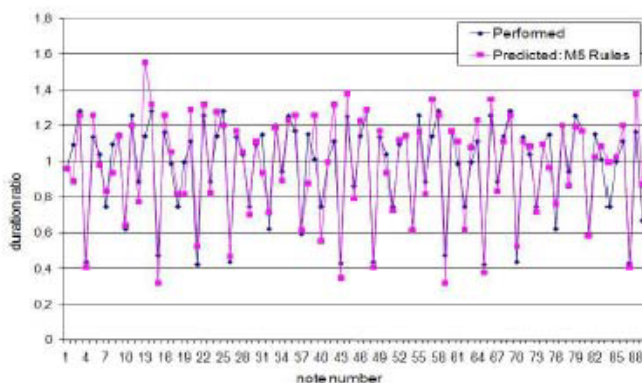
### 4.3. Algorithm

We apply Tilde's top-down decision tree induction algorithm ([1]). Tilde can be considered as a first order logic extension of the C4.5 decision tree algorithm: instead of testing attribute values at the nodes of the tree, Tilde tests logical predicates. This provides the advantages of

both propositional decision trees (i.e. efficiency and pruning techniques) and the use of first order logic (i.e. increased expressiveness). The musical context of each note is defined by predicates context and narmour. context specifies the note features described above and narmour specifies the Narmour groups to which a particular note belongs, along with its position within a particular group. Expressive deviations in the performances are encoded using predicates stretch and dynamics. stretch specifies the stretch factor of a given note with regard to its duration in the score and dynamics specifies the mean energy of a given note. The temporal aspect of music is encoded via the predicates pred and succ. For instance, succ(A,B,C,D) indicates that note in position D in the excerpt indexed by the tuple(A,B) follows note C.

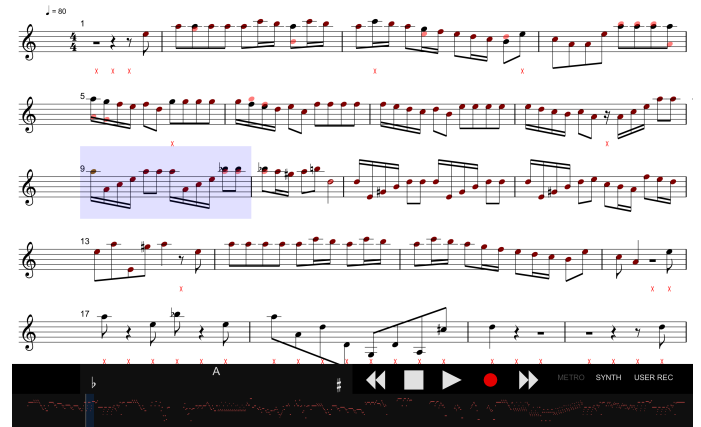
#### 4.4. Results

We evaluated the expressive performance model obtaining correlation coefficients of 0.88 and 0.83 for the duration transformation and note dynamics prediction tasks, respectively. These numbers were obtained by performing 10-fold cross validation on the training data. The induced model seem to capture accurately the expressive transformations the musician introduce in the performances. Figure 1 contrasts the note duration deviations predicted by the model and the deviations performed by the violinist.



**Figure 1: Note deviation ratio for a tune with 89 notes. Comparison between performed and predicted by the expressive performance model**

We have implemented a prototype, which allows students to visualize the score, the expressive performance generated by the computational model, and their own performance. The prototype allows students to compare their performances with the target performance in terms of duration, and/or energy in real-time.



**Figure 2: Visualization prototype**

#### 5. Conclusion

We applied machine learning techniques to learn computational models of music expression in violin performances. The aim is to use this models in a music learning prototype for teaching students how to play expressively. The induced models seem to capture accurately the expressive transformations the musician introduce in the performances. The implication of this work is that its outcome has the potential to contribute to the engagement of musicians in the community by making more appealing music practice and instrument training.

#### References

- [1] H. Blockeel, L. D. Raedt, and J. Ramon. Top-down induction of clustering trees. In *Proceedings of the 15th International Conference on Machine Learning*, 1998.
- [2] Bresin, R. (2000). *Virtual Virtuosity: Studies in Automatic Music Performance*. PhD Thesis, KTH, Sweden.
- [3] Friberg, A.; Bresin, R.; Fryden, L.; 2000. *Music from Motion: Sound Level Envelopes of*



Tones Expressing Human Locomotion. *Journal of New Music Research* 29(3): 199-210.

[4] Gabrielsson, A. (1999). The performance of Music. In D.Deutsch (Ed.), *The Psychology of Music* (2nd ed.) Academic Press.

[5] Klapuri, A. (1999). Sound Onset Detection by Applying Psychoacoustic Knowledge, *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*.

[6] Lopez de Mantaras, R. and Arcos, J.L. (2002). AI and music, from composition to expressive performance, *AI Magazine*, 23-3.

[7] Maher, R.C. and Beauchamp, J.W. (1994). Fundamental frequency estimation of musical signals using a two-way mismatch procedure, *Journal of the Acoustic Society of America*, vol. 95 pp. 2254-2263.

[8] McNab, R.J., Smith L.I. A. and Witten I.H., (1996). *Signal Processing for Melody Transcription*, SIG working paper, vol. 95-22.

[9] Mitchell, T.M. (1997). *Machine Learning*. McGraw- Hill.

[10] Narmour, E. (1990). *The Analysis and Cognition of Basic Melodic Structures: The Implication Realization Model*. University of Chicago Press.

[11] Rafael Ramirez, Amaury Hazan, Esteban Maestre, Xavier Serra, *A Data Mining Approach to Expressive Music Performance Modeling*, in *Multimedia Data mining and Knowledge Discovery*, Springer.

[12] Repp, B.H. (1992). Diversity and Commonality in Music Performance: an Analysis of Timing Microstructure in Schumann's *Traumerei*. *Journal of the Acoustical Society of America* 104.

[13] Saunders C., Hardoon D., Shawe-Taylor J., and Widmer G. (2004). Using String Kernels to Identify Famous Performers from their Playing Style, *Proceedings of the 15th European Conference on Machine Learning (ECML'2004)*, Pisa, Italy.

[14] Seashore, C.E. (ed.) (1936). *Objective Analysis of Music Performance*. University of

Iowa Press.

[15] Serra, X. and Smith, S. (1990). Spectral Modeling Synthesis: A Sound Analysis/Synthesis System Based on a Deterministic plus Stochastic Decomposition, *Computer Music Journal*, Vol. 14, No. 4.

[16] Stamatatos, E. and Widmer, G. (2005). Automatic Identification of Music Performers with Learning Ensembles. *Artificial Intelligence* 165(1), 37-56.

[17] Todd, N. (1992). The Dynamics of Dynamics: a Model of Musical Expression. *Journal of the Acoustical Society of America* 91.

# The Million Playlists Songs Dataset: a descriptive study over multiple sources of user-curated playlists

Felipe Falcão<sup>1\*</sup>, Daniel Mélo<sup>1†</sup>

<sup>1</sup>Laboratory of Distributed Systems – Federal University of Campina Grande  
Av. Aprígio Veloso, s/n, Bloco CO – 58.429-900, Campina Grande, PB

felipev@lsd.ufcg.edu.br, danielgondim@lsd.ufcg.edu.br

## Abstract

User interest for playlists is increasing as current music streaming services become more and more popular. In order to get sets of songs that best match current musical needs (e.g. size, diversity, mood), one has to select a compatible playlists source between a representative number of options. Most of available music streaming platforms (e.g. Spotify, Pandora, Deezer) already contain playlists searching mechanisms, but as a secondary source of such information we have websites that allow users to submit, manage and publish their own playlists, organizing them according to some specific criteria. This paper proposes a descriptive study over four of these websites in such way that it categorize the groups of playlists available on each one. By recursively crawling and querying data from these sources and enriching it with high-level acoustic information fetched from AcousticBrainz, we were able to build a dataset called *Million Playlists Songs Dataset* which guided the descriptive process and is now available for further investigation.

## 1. Introduction

Nowadays, with the spread of music streaming services such as Spotify<sup>1</sup>, Pandora<sup>2</sup>, Google Play Music<sup>3</sup>, Deezer<sup>4</sup> and etc., where it is possible to find millions of songs quickly and easily, it is quite common for users to organize their music creating or searching for playlists that suit their current musical needs (mood, size, genre, artist, diversity, etc.).

\*Supported by CAPES.

†Supported by CAPES.

<sup>1</sup><https://www.spotify.com/>

<sup>2</sup><http://www.pandora.com>

<sup>3</sup><https://play.google.com/music>

<sup>4</sup><https://www.deezer.com>

For being such a common concept nowadays in music, playlists have become an important object of study for the Music Information Retrieval (MIR) area. One of the great challenges involved in this study is to understand the user behavior when creating playlists. Is there a preservation in the songs features, or they prefer a heterogeneity? Is there any change in the choice of songs according to the context in which the playlist was created? As can be seen, many variables are involved in this activity, making this analysis quite complex for the MIR area.

In order to help resolve these questionings, this paper proposes a brief descriptive analysis of four playlists data sources. These sources are websites that allow users to create, manage, and share playlists manually. These sites are: 8tracks, Art of the Mix, Playlists.net and *Vagalume* (a description of these sites can be found in section 3.1). This analysis seeks to categorize groups of playlists, identifying common characteristics that may indicate user preferences.

And as a second contribution, this work also establishes the creation of a new data source, composed of all the data of the four websites analyzed, but also enriched with high-level acoustic characteristics of the songs. Such information could be retrieved using the MusicBrainz<sup>5</sup> and AcousticBrainz<sup>6</sup> platforms. Thus, as far as we know, we provide the community with a totally innovative dataset, containing not only songs from playlists, but also their high-level acoustic features.

This paper is structured as follows: in section 2 we have identified some related works, and also emphasize the novelties of this work. Section 3

<sup>5</sup><https://musicbrainz.org/>

<sup>6</sup><https://acousticbrainz.org/>

describes how the dataset used in this work was developed. All the descriptive analysis of the dataset can be found in section 4. And finally, in section 5, we discuss about the conclusions of this work as well as possible future work.

## 2. Related Work

It is notable that with the spread of streaming music services, such as Spotify, Deezer, Pandora, etc., the amount of music available to users has increased significantly. To keep up with this increase and improve the user experience, multiple platforms provide the ability to create and share playlists

We can divide the activity of creating playlists into two large groups: (i) automatic generation and; (ii) manual. This first group has already been well explored in research in the area of MIR as reviewed by Bonnin and Jannach [1], indicating ways for automation based on user's listening habits [2], grouping songs by similarity of high level characteristics [3], user real-time physiological feedback [4], as well as similarity of their frequency spectrum [5]. The manual generation of playlists requires further investigation by researchers, since it is necessary to understand the behavior of users when creating playlists. Some steps have already been taken in this direction [6], in addition there are also works that examine corpus of playlists created manually [7], but these works are still vague.

Besides that, there are few datasets that incorporate data from playlists created manually. We can cite the Art of The Mix, made available by McFee and Lancriet [8], #nowplaying [9] and 30Music [10]. Although they are datasets with a considerable amount of data, they are restricted only to common descriptions such as album name and tags (in addition to the song and artist name).

This work fills this gap since it proposes the creation of a huge dataset, with almost 2 million musical entries of playlists, where not only the common features are stored, but also the high-level acoustic ones of such songs. These features are obtained through AcousticBrainz [11].

## 3. The Dataset

The *Million Playlists Songs Dataset* - MPSD (deliberately a tribute to the *Million Song Dataset* [12], whose work guided us during our efforts) comprises data fetched from four different sources of user-curated playlists. Since most of current studies [10, 13] already considered monitoring broadly used platforms such as Twitter, Last.FM and Spotify, we have, on the other hand, focused on crawling data from secondary platforms which, although not holding as much users as the aforementioned systems, also remains as an unexplored source of playlists data, with a representative number of enthusiasts and curated playlists.

### 3.1. Data Sources

The methodology applied during the preparation of our dataset is hybrid and featured by both crawling and querying approaches. The choice between some of these approaches was defined by how the desired data was structured on their websites and how easy it would be for authors to have the maximum of data available for analysis in the shortest amount of time.

The first source of playlists data included on the dataset is the *Vagalume* website. *Vagalume*<sup>7</sup> is a music portal created in Brazil on 2002, initially conceived as a public database of song's lyrics. As years went by and the platform received more attention by the community (specially from Brazil and Portugal), features were expanded and users were then allowed to upload public content, such as public playlists composed by Youtube music videos. All the dataset *Vagalume*-related data was fetched by crawling playlists pages from the main *Vagalume* profile<sup>8</sup> and all playlists from his respective followers, totaling 35,600 profiles. Profiles without registered playlists were ignored.

The proposed dataset also comprises data coming from the playlists.net<sup>9</sup> website. By also crawling playlists pages from this platform we have enriched our dataset with playlists hosted on Spotify and submitted to this platform by

<sup>7</sup><https://www.vagalume.com.br>

<sup>8</sup><https://meu.vagalume.com.br/sitevagalume>

<sup>9</sup><http://playlists.net/>

a very active community of users interested on discovering playlists that sometimes cannot be found directly on Spotify browser. Although site creators claim to have about 170,000 registered playlists, many of them are not available on Spotify anymore and some others were not listed on the webpage at crawling time.

A well-known platform was used as data source for our work: 8tracks.com<sup>10</sup>. Founded in 2006, 8tracks is a collaborative platform that allows users to share and discover music in a simple, legal, and free way. On this platform, users can create and share playlists with at least eight songs. The data from this platform was provided to us directly by its administrator.

Finally, we also use the data from the Art of the Mix website<sup>11</sup>. This site integrates playlists created on iTunes, nightly. These data were provided by McFee and Lancriet [8] and contains information from more than 100,000 playlists.

### 3.2. Crawling

In order to automate the process of recursively looking into multiple sections of several websites, the process of building MPSD was aided by some computer-aided software engineering (CASE) tools that provided us some ready-to-use features without which any of the now-available artifacts would be published in time.

Crawlings were fully accomplished by running Python scripts along with Scrappy<sup>12</sup> tasks that recursively visited thousands of webpages to fetch desirable data stored on nested tags of HTML documents. Scrappy is a fast high-level web crawling open source framework written in Python to assist developers on tasks based on the extraction of structured data from websites and APIs. By providing mechanisms of recursive crawlings, this tool allows users to start looking at specified URLs, extract desired information present on the HTML document and search for external links this page might have to proceed with the crawling loop as deep as planned. The result of this process is a list of visited pages as well as the set of extracted data.

<sup>10</sup><https://8tracks.com/>

<sup>11</sup><http://www.artofthemix.org/>

<sup>12</sup><https://scrappy.org>

Besides, to simulate user-specific behavior (link clicks, in this case) scripts were also enriched with Selenium<sup>13</sup> features. Even though originally designed for software-testing tasks, Selenium suite was able to provide us some web-browser automation tools that helped us to load some data that could only be available by interacting with web interface elements via link clicks, since Scrappy isn't currently able to perform such kind of operation.

All these aforementioned tools were combined in order to create a powerful and generic web crawler that initially ran over the two chosen web-based sources (*Vagalume* and *Playlists.net*) at the same time in a single computer<sup>14</sup> to produce the expected outputs. These tasks took about eight uninterrupted days to be finished using our available infrastructure.

### 3.3. Extraction

Due to the large amount of data and the limitation of time and resources for processing, a reduction in the amount of data used from 8tracks and Art of the Mix was required.

For 8tracks, we only extracted from the database playlists with more than ten songs. Due to inconsistencies in the given database, several playlists with fewer than ten songs were returned, as their *tracks\_count* attribute indicated a value greater than 10, but had a smaller list of songs. As this inconsistency would not cause any damage to our work, we consider all the data returned in this extraction, which counted a total of 27,606 playlists, dated from September 2007 until June 2012.

32,681 playlists were extracted from Art of the Mix. This value was reached after we left an extraction script running for 7 uninterrupted days with the resources we had. These playlists cover the period from January 1998 to June 2011.

### 3.4. Acoustic Enrichment

As our crawling and querying techniques were extracting playlists and tracks metadata

<sup>13</sup><http://www.seleniumhq.org>

<sup>14</sup>In our experiments, we used an 8-core Intel(R) Core(TM) i5-4200U CPU @ 1.60GHz with 8GB of RAM memory running Ubuntu 16.04.2 LTS. All our codes were implemented and executed using Python 2.7.

available on all of the chosen sources, we also tried to enrich even more the gathered information by appending to it some extra high-level acoustic features available for querying on AcousticBrainz database. For so, every single track on each of the studied sources was queried on MusicBrainz so it could have assigned to itself a MusicBrainz Identifier (MBID) which would be used to fetch acoustic data on AcousticBrainz, if available.

Since AcousticBrainz is a recent platform and also taking into consideration that the queried songs came from secondary sources we could not get all the information planned. Instead, we realized that only 10,45% of our comprised songs were able to be enriched with acoustic data. Even though it is less than half of all dataset songs, we consider that it is a representative result for this first effort.

#### 4. Dataset Analysis

MPSD is currently a collection of 1,993,607 tracks of 74,996 distinct playlists songs annotated with 45 field descriptors (both statistics of each source and all field descriptors can be examined on Tables 1 and 2, respectively). Altogether, 617,242 distinct track names of 221,560 distinct artists can be fully analyzed in a 576,6MB CSV file available on the project GitHub repository <sup>15</sup>.

**Table 1: Playlists statistics**

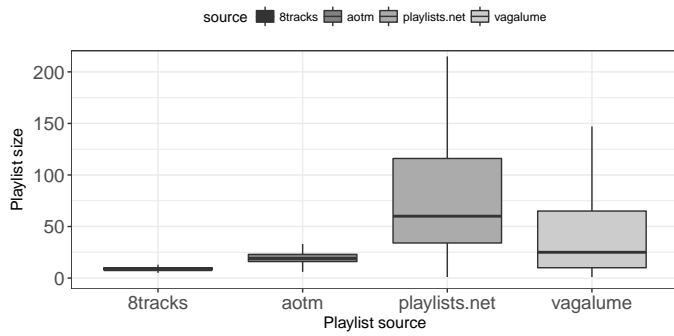
Sources	# playlists	# tracks	# artists	Maximum Playlist Size
AotM	32,681	296,344	110,154	60
8tracks	27,606	115,660	50,354	226
Vagalume	9,584	124,627	6,611	2995
Playlists.net	5,125	185,291	92,236	7287

Moving forward with our analysis and deeply looking into our data we could elaborate some hypothesis about the size of our available playlists. Table 1 shows us that both *Vagalume* and Playlists.net have a smaller amount of distinct playlists stored on database, but the total crawled songs for each source (646,070 and

<sup>15</sup><https://github.com/felipevieira/computacao-e-musica-lsd/tree/master/sbcm-2017>

**Table 2: All song's field descriptors comprised by MPSD**

Field	Description
source	Source that hosts the playlist (Possible values: <i>Vagalume</i> , AoTM, 8tracks, playlists.net)
user_id	An unique playlist identifier (format varies from source to source)
track_name	Song title
artist_name	Artist or band that performs that specific version of a song
mbids	List of MusicBrainz identifiers (a 36 character Universally Unique Identifier that is permanently assigned to each entity in the database)
playlist_id	An unique playlist identifier (format varies from source to source)
tags	List of labels attached to each song in order to provide extra-information about it (format varies from source to source)
playlist_name	Playlist title
danceability_value/prob	Danceability value and probability as defined by the Essentia classifier model [14] (Possible values: danceable, not_danceable)
gender_value/prob	Gender value and probability as defined by the Essentia classifier model [14] (Possible values: male, female)
genre_[dataset]_value/prob	Genre value and probability as defined by the Essentia classifier model [14] for four different datasets
ismir04_rhythm_value/prob	Rhythm value and probability as defined by Goyiyon classifier model [15]
mood_[type]_value/prob	Mood value and probability as defined by the Essentia classifier model [14] for eight different mood types
timbre_value/prob	Timbre value and probability as defined by the Essentia classifier model [14] (Possible values: bright, dark)
tonal_atonal_value/prob	Tonal/Atonal value and probability as defined by the Essentia classifier model [14] (Possible values: tonal, atonal)
voice_instrumental_prob/source	Voice/Instrumental value and probability as defined by the Essentia classifier model [14] (Possible values: voice, instrumental)



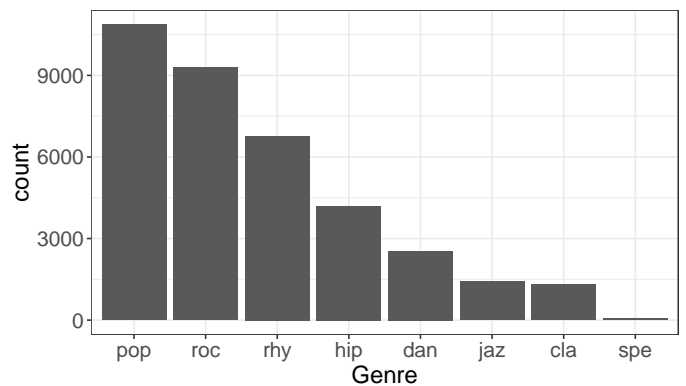
**Figure 1: Playlists size distribution**

432,351 tracks for these two sources, respectively, against 643,349 and 258,727 from 8tracks and Art of The Mix) does not reflect this minority. By checking average and standard deviation information for grouped data (Table 3) in addition with the playlists size distribution on Figure 1 we confirm our theory, concluding that even though we were able to crawl more playlists from AotM and 8tracks, the ones obtained from *Vagalume* and *Playlists.net* had more songs on them.

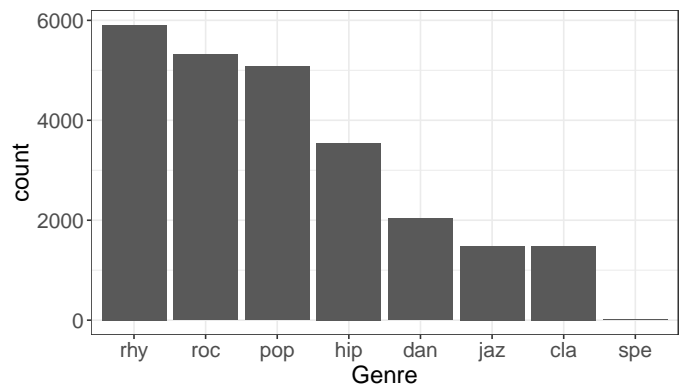
Since genre can be an important factor considered by users when searching for playlists, this study also tried to find out and understand the distribution of genres along the tracks fetched on each of our sources. Such task was aided by AcousticBrainz information added to our dataset on the Acoustic Enrichment Phase mentioned on subsection 3.4. The field used to best summarize genre information about a song was the *genre\_rosamerica\_value*, which estimates a song genre by using the Rosamerica Collection [16, 17] while training a classifier model that assigns one of the eight possible genre values (rhythm & blues, rock, pop, hip-hop, dance, jazz, classic and speech) to new song entries.

Our analysis shows some concrete differences between genres distributions over all four sources. While 8tracks and AotM users seems to have very similar affinities with rock and r&b (in the same order), *Vagalume* and *Playlists.net* users (Figures 2 and 3, respectively) prefer pop and r&b songs, respectively.

In an effort to exemplify some useful applications of MPSD on what regards a better understanding of how users behaviour when creat-



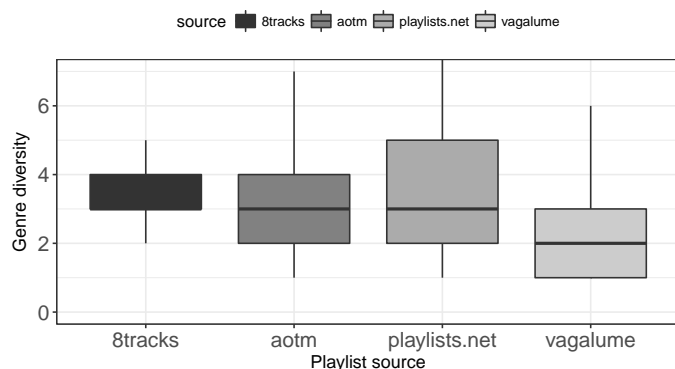
**Figure 2: Genre Histogram of Vagalume's playlists**



**Figure 3: Genre Histogram of Playlists.net's playlists**

**Table 3: Statistical data about playlists size**

Sources	Average	Standard Deviation
AotM	19.68 tracks	6.22
8tracks	9.37 tracks	4.53
<i>Vagalume</i>	67.41 tracks	148.12
Playlists.net	84.13 tracks	96.78
<b>Total</b>	29.35 tracks	66.68



**Figure 4: Playlists genre diversity distribution**

ing playlists differ from one source to another, we have used some of the data available on the dataset to extract some genre-diversity insights observed on the crawled playlists. For this, we summarized our dataset to check how many distinct genres were present in each of our comprised playlists. Since Acoustic Enrichment Phase wasn't able to fetch high-level acoustic data for all playlist songs, in order to minimize the effect of this lack of data we have filtered our dataset to only consider playlists with more than five songs contemplated with acoustic data (i.e. *genre\_rosamerica\_value* field descriptor). The result of our analysis can be examined on Figure 4 concluding that *Vagalume* is the source with the less genre-diversified playlists (with a median of two different genres per playlist), while *playlists.net* sets of songs can be considered as the most diverse in terms of genre, even though its diversity median was the same as *8tracks* and *AoTM*.

## 5. Further Work

In the course of this paper we were able to present all the methodology applied on the composition of the Million Playlists Songs Dataset: a now-public dataset of metadata information regarding playlists songs from four web platforms designed to allow user curation over playlists. Besides, as a way of exemplify studies that may be conducted over this set of data, a simple descriptive analysis was performed over all data to extract insights about the distinct sources of data considered when building the dataset.

There is plenty of contributions to be performed in order to complement this preliminary study. One of them is the gradual increment of this dataset with all kinds of sources and information that could not be contemplated by this current research by time and computational reasons. By conducting a long-term research, one can increment this dataset with new sources and crawl older data information, and these would be good approaches to attach even more value to this dataset.

In addition, a more detailed study of the data in this dataset could be carried out in order to extract more information about the behavior of the users during the process of creating playlists, e.g. identify the features that matter most to the users, the trends of the users, etc.

Another gap found during the current study is the lack of a complete state-of-art framework designed to extract acoustic features from as much songs as possible. Even with the arise of *AcousticBrainz* (with almost 5 million registered songs) we still faced minor issues when trying to fetch acoustic data from *MPSD* tracks, specially on what refers to brazilian music.

## References

- [1] Geoffroy Bonnin and Dietmar Jannach. Automated generation of music playlists: Survey and experiments. *ACM Comput. Surv.*, 47(2):26:1–26:35, November 2014.
- [2] Andreja Andric and Goffredo Haus. Automatic playlist generation based on tracking user's listening habits. *Multimedia Tools and Applications*, 29(2):127–151, 2006.
- [3] Steffen Pauws and Berry Eggen. Pats: Realization and user evaluation of an automatic playlist generator. In *ISMIR*, 2002.
- [4] Nuria Oliver and Lucas Kreger-Stickles. Papa: Physiology and purpose-aware automatic playlist generation. In *ISMIR*, volume 2006, page 7th, 2006.
- [5] Beth Logan. Content-based playlist generation: Exploratory experiments. In *ISMIR*, 2002.
- [6] Sally Jo Cunningham, David Bainbridge, and Annette Falconer. "more of an art

- than a science”: Supporting the creation of playlists and mixes. 2006.
- [7] Dietmar Jannach, Iman Kamehkhosh, and Geoffray Bonnin. Analyzing the characteristics of shared playlists for music recommendation. In *RSWeb@ RecSys*, 2014.
- [8] Brian McFee and Gert R. G. Lanckriet. Hypergraph models of playlist dialects. In *ISMIR*, 2012.
- [9] Martin Pichl, Eva Zangerle, and Günther Specht. Towards a context-aware music recommendation approach: What is hidden in the playlist name? In *Data Mining Workshop (ICDMW), 2015 IEEE International Conference on*, pages 1360–1365. IEEE, 2015.
- [10] Roberto Turrin, Massimo Quadrana, Andrea Condorelli, Roberto Pagano, and Paolo Cremonesi. 30music listening and playlists dataset. In *RecSys Posters*, 2015.
- [11] Alastair Porter, Dmitry Bogdanov, Robert Kaye, Roman Tsukanov, and Xavier Serra. Acousticbrainz: a community platform for gathering music information obtained from audio. In *International Society for Music Information Retrieval Conference (ISMIR’15)*, 2015.
- [12] Thierry Bertin-Mahieux, Daniel PW Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *ISMIR*, volume 2, page 10, 2011.
- [13] Eva Zangerle, Martin Pichl, Wolfgang Gassler, and Günther Specht. #nowplaying music dataset: Extracting listening behavior from twitter. In *Proceedings of the First International Workshop on Internet-Scale Multimedia Management*, pages 21–26. ACM, 2014.
- [14] Dmitry Bogdanov, Nicolas Wack, Emilia Gómez, Sankalp Gulati, Perfecto Herrera, Oscar Mayor, Gerard Roma, Justin Salamon, José R Zapata, Xavier Serra, et al. Essentia: An audio analysis library for music information retrieval. In *ISMIR*, pages 493–498, 2013.
- [15] Fabien Gouyon. Dance music classification: A tempo-based approach. 2004.
- [16] Dmitry Bogdanov, Alastair Porter, Perfecto Herrera, and Xavier Serra. Cross-collection evaluation for music classification tasks. In *Proc. of the Int. Conf. on Music Information Retrieval (ISMIR)*, 2016.
- [17] Enric Guaus i Termens. *Audio content processing for automatic music genre classification: descriptors, databases, and classifiers*. PhD thesis, Universitat Pompeu Fabra, Barcelona Barcelona, 2009.



# Timbre spaces with sparse autoencoders

Pablo E. Riera<sup>1</sup>, Manuel C. Eguía<sup>1</sup>, Matías Zabaljáuregui<sup>2</sup>

<sup>1</sup>Laboratorio de acústica y percepción sonora  
Escuela Universitaria de Artes, Universidad Nacional de Quilmes  
Roque Sáenz Peña 352, Bernal Buenos Aires, Argentina

<sup>2</sup>Laboratorio de Inteligencia Artificial Aplicada  
Departamento de Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires  
Pabellón I, Ciudad Universitaria, Ciudad de Buenos Aires, Argentina.

pablo.riera@gmail.com

## Abstract

Timbre perception studies emphasize the multidimensional nature of timbre and many rely on dimensionality reduction techniques to visualize perceptual similarity evaluations or sound descriptors that encompass timbre perception. In this work, we explore the uses of sparse autoencoders to perform unsupervised learning and nonlinear dimensionality reduction to extract a spectral code representation that is used for timbre analysis and visualization. Using only one music fragment in the autoencoder learning process generates an overfitted reconstruction but gives a low dimensional neuronal activity pattern which encodes all the sound spectrum information and could be used for synthesis as a neuronal music score.

## 1. Introduction

Timbre is widely recognized as a highly complex and multidimensional percept that cannot be (or hardly can be) accounted in terms of a few quantitative descriptors [1][2]. The process of timbre perception is closely related to the tasks of sound identification and classification and is concomitant to the hierarchical organization of sensory systems [3]. A common approach to tackle this issue in computational studies consists of calculating some sound descriptors relevant to timbre perception, such as Mel-frequency Cepstral Coefficients (MFCC) [4] or Spectral Contrast [5] and then applying dimensionality reduction techniques for visualization, as multidimensional scaling [6], isomaps [7] and self-organized maps [8], among others.

On the other hand, recent advances in deep neural networks, in which several layers of nodes are used to build up progressively more abstract representations of the inputs, have contributed to develop a new approach to this problem, yielded promising results in music related tasks such as instrument classification [9][10], timbre analysis [11], genre classification [12][13], among others, [14][15] and sample based sound synthesis [16]. Several architectures and learning procedures have proven successful at processing audio data, in particular, we can mention recent uses of autoencoders in audio synthesis [17][18], statistical parametric speech synthesis [19], adaptive reduced-dimensionality equalization [20], and denoising and dereverberation [21]. As an alternative to autoencoders, linear methods like sparse coding and nonnegative matrix factorization have also been used for sound classification [22] and source separation [23].

Autoencoders are a type of neural network having a coding stage and a decoding stage, in such a way that a bottleneck is generated in the middle layer (see Figure 1). Usually, the learning strategy involves minimizing the difference between the input and the coded-decode output. This learning procedure could be used for nonlinear dimensionality reduction [24] or nonlinear PCA [25][26].

In this work, we explore the use of sparse autoencoders with spectral inputs for efficiently learning timbre representation and synthesis, in a similar fashion to [18]. In contrast to this work, we perform a complete unsupervised learning with one musical or audio fragment at a time. This generates a network that overfits the audio

fragment used, and focus on nonlinear dimensionality reduction. In addition, a sparse penalty is included in the cost function in order to favor a sparser representation in the code layer. When using the system for sound synthesis, the spectral representation of the input needs to be invertible and bear an appropriate method for phase reconstruction [27].

In the next section, we describe the autoencoder structure and the learning methods. In the results section, we compare timbre spaces generated with the autoencoder code layer to those generated with MFCC and Spectral Contrast as descriptors.

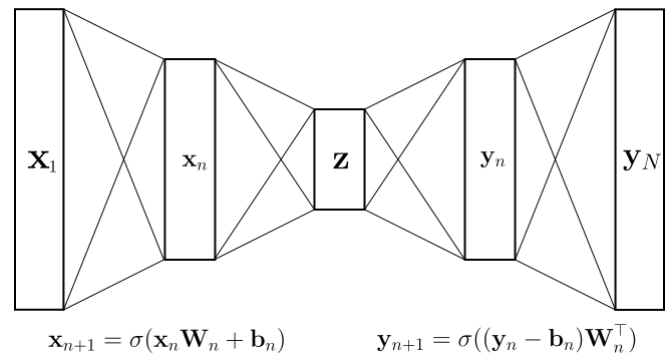
## 2. Methods

The log-magnitude power spectrogram of a monaural audio fragment was used as input, after resampling it to a lower sampling rate (in order to make the computations lighter) and computing the short-time Fourier transform (STFT) with fixed window and step length. Every STFT window was considered as a data sample (number of observations), thus the number of frequency bins was equal to the number of input neurons (number of features).

The autoencoder was built with tied weights in the coding and decoding stages. The numbers of layers and neurons per layer were explored manually before adopting a simplified structure in which the number of neurons of each layer was reduced by a factor of two until a minimum number is obtained in the code layer, and for the decoding stage, the numbers are increased by the same factor. This architecture allowed a reasonable learning performance. In concrete, the number of neurons per layer finally adopted was  $2^k$   $k = 10, 9, 7, 5, 3, 5, 7, 9, 10$ , hence the code layer has only 8 neurons.

Several activation functions were tested. The results shown here were obtained using a soft-plus function and min max normalization. Similar results were obtained with tanh function and z-score normalization.

The cost function consisted of three terms:



**Figure 1: Autoencoder layout.**  $X_n$  represents the activities on the coding stage  $Z$  represents the code layer activity, and  $Y_n$  the activities on the decoding stage. A clear bottleneck is visualized.

$$\begin{aligned}
 E = & \frac{1}{2N_0} \sum_{n=1}^{N_0} (X_n - Y_n)^2 \\
 & - \frac{\alpha}{N_z} \sum_{n=1}^{N_z} \hat{Z}_n \log(\hat{Z}_n^2) \\
 & + \lambda \frac{1}{2K} \sum_{k=1}^K (\|W_k\|_2^2 + \|b_k\|_2^2) \quad (1)
 \end{aligned}$$

The first term corresponds to the averaged squared difference between the input  $X$  and the output  $Y$  of the autoencoder.  $N_0$  denotes the size of input and output layers. The second term corresponds to a sparse regularization component, provided by the entropy of the normalized activity of the code layer  $\sum(\hat{Z}) = 1$ .  $N_z$  stands for the code layer size. Finally, the third term includes a  $L2$  regularization on the weights and bias.  $K$  corresponds to the number of weight matrices.

Before training the complete autoencoder, a pretraining [24] stage was performed on single hidden layer autoencoders, where the input of a single layer of the autoencoder was the code layer activity of the previous one. The network was optimized with Adam method [28].

After the training, the analysis consisted on inspecting the activity of the code layer. This activity plus the weights and bias contain all the information for reconstructing the original audio signal. In a general sense, this neuronal activity can be put into correspondence to a *timbre score*

Parameters	Values
Sampling frequency	22050
Window size	1024 (46 ms)
Step size	256 (11 ms)
Number of spectrum samples	5168
Layers dimensions	$2^k$ $k = 10, 9, 7, 5$ $3, 5, 7, 9, 10$
Activation function	softplus
$\alpha$	0.8
$\lambda$	0.005
Learning rate	0.005
Batch size	200

**Table 1: Hyper-parameters of the sparse autoencoder**

of the musical fragment used as input, or a trajectory on the timbre space of the audio signal.

For visualization of the timbre space, reduction of the dimensionality of the activations of the code layer was done by principal components analysis (PCA). Finally, for the sonification of the reconstructed spectrogram, we used the original phase information. This was done with the aim of preserving the audio quality, but phase reconstruction algorithms could be used as well.

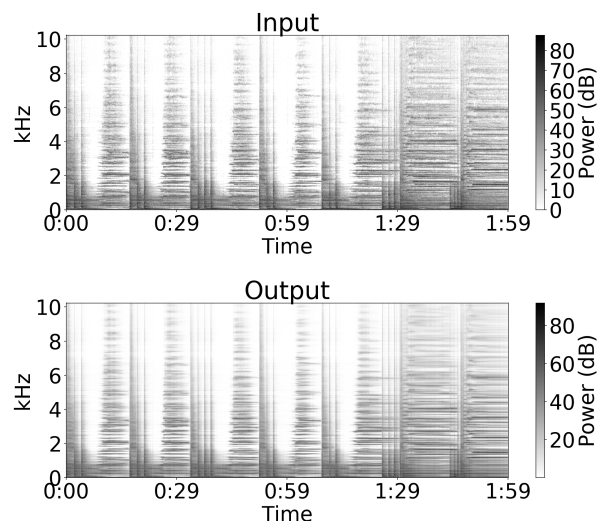
### 3. Results

The results displayed here are obtained using the first 7 cycles of Grisey’s *Partiels* [29] as input. We have chosen this example because, in one hand, this piece has an outstanding historic relevance in the development of timbre in 20th-century music, and on the other hand, it behaves as a good data set, mainly due to the fact that is basically composed by repetitions of similar sounds with variations.

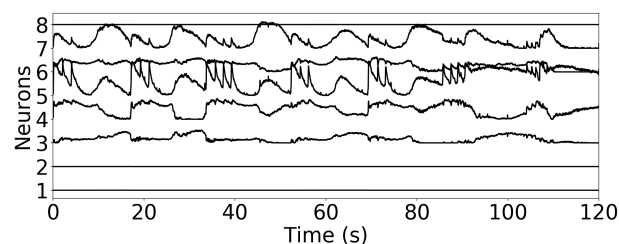
The hyper-parameters of the autoencoder used for this results are shown in table 1.

In figure 2, we show the spectrogram of the original audio signal along with the reconstruction obtained as output. Despite being a highly detailed reconstruction, the output displays fewer variations than the original, due to the fact that the autoencoder has some denoising properties.

In figure 3 we display the neural activity pat-



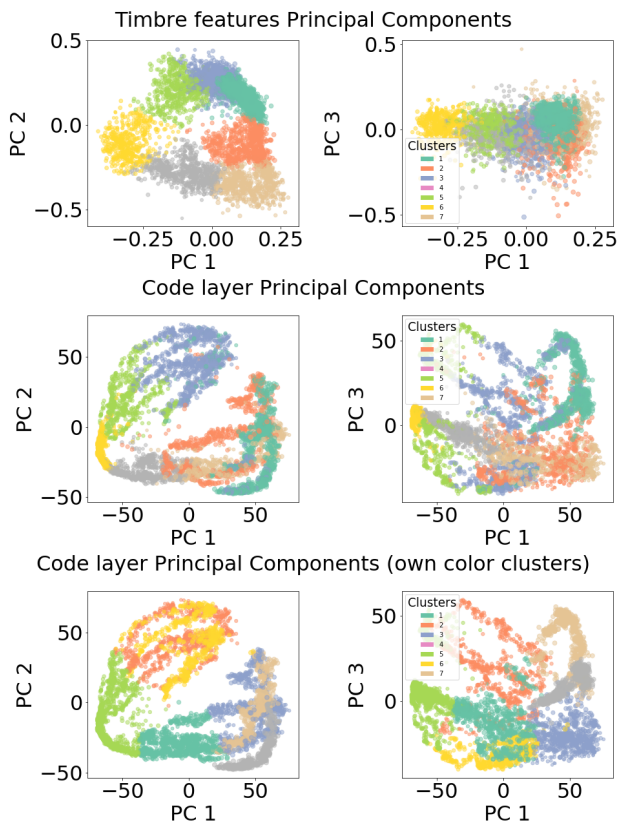
**Figure 2: Top: Input data spectrogram of a fragment of Grisey’s *Partiels*. Bottom: the reconstructed spectrogram by the autoencoder.**



**Figure 3: Code layer activities. As the reconstruction is almost identical to the original sound, we could interpret this activities as the music neural score.**

tern of the code layer. The sparse regularization cost forces a sparse activity pattern, and some neurons ended with null activation. Nonetheless, different runs of the optimization could end with different activation patterns (and same overall cost), due to random initial weights and bias.

Figure 4, shows on the top row, the timbre space generated by three principal components of standard timbre descriptors (in this case a combination of 20 MFCC and 7 Spectral Contrast coefficients). The color shades were obtained by k-means clustering on those descriptors. Seven clusters were used to guide the visual inspection of both timbre spaces and to match the different regions. In the middle row are the



**Figure 4: Top and middle: Timbre space generated by the first three principal components of a set of standard timbre descriptors (MFCC, Spectral Contrast) and by the code layer activities. Bottom: Same data as in the middle plot, but with different clustering colors (see text for a full description).**

PCA components for the code layer activity pattern with the same color clustering. The bottom row uses its own clusters instead. It could be observed that some clusters remain close in both top and middle row timbre spaces, but there are differences with those clusters generated by the code layer in the bottom plot, more structure arise. This is reflected also in the difference between the third PCA component from the timbre features and the third component of the code layer. The standard timbre descriptors elicit a more compacted space than from the autoencoder which has a torn structure.

In the first 7 cycles of *Partiels*, an alternating pattern is produced between the trombone and contrabass low notes and the strings and woods

high notes that generate a cyclic trajectory in the timbre space which is present in both timbre spaces (timbre features and code layer).

## 4. Conclusions

We presented a computational timbre analysis method involving a sparse autoencoder. When using this type of neural networks, the code layer is able to learn a meaningful representation that is capable of reconstructing the original input data, the sound spectrum in the present work. The activity of the code layer is used for timbre analysis and it behaves like an audio specific, a small set of sound descriptors.

The results presented here focused on learning from a single sound or music fragment, but the same approach could be expanded to learning from a corpus of music fragments. Different network architectures may be necessary and also a different learning paradigm like classification [18].

Regarding the synthesis capabilities of the system, when trained with a music fragment, we consider the code layer activities as a neural score that interprets the music. This could be useful for timbre-oriented audio processing and source separation.

Generated timbre spaces by the autoencoder shown more structure than those generated by standard timbre descriptors. For a complete analysis of the timbre space, a perceptual measurement or an exhaustive listening task is needed by comparing the clusters and their relative locations.

## 5. References

### References

- [1] Stephen McAdams. Musical timbre perception. *The psychology of music*, pages 35–67, 2013.
- [2] Carol L Krumhansl. Why is musical timbre so hard to understand ? In *Structure and perception of electroacoustic sound and music*, volume 9, pages 43–53. 1989.

- [3] Kailash Patil, Daniel Pressnitzer, Shihab Shamma, and Mounya Elhilali. Music in our ears: the biological bases of musical timbre perception. *PLoS Comput Biol*, 8(11):e1002759, 2012.
- [4] Beth Logan. Mel Frequency Cepstral Coefficients for Music Modeling. *International Symposium on Music Information Retrieval*, 28:11p., 2000.
- [5] Dan-Ning Jiang, Lie Lu, Hong-Jiang Zhang, Jian-Hua Tao, and Lian-Hong Cai. Music type classification by spectral contrast feature. In *Multimedia and Expo, 2002. ICME'02. Proceedings. 2002 IEEE International Conference on*, volume 1, pages 113–116. IEEE, 2002.
- [6] John M. Grey. Multidimensional perceptual scaling of musical timbres. *The Journal of the Acoustical Society of America*, 61(5):1270–1277, 1977.
- [7] John Ashley Burgoyne and Stephen McAdams. Non-linear scaling techniques for uncovering the perceptual dimensions of timbre. In *ICMC*, 2007.
- [8] MA Loureiro, HB de Paula, and HC Yehia. Timbre Classification Of A Single Musical Instrument. *ISMIR*, 2004.
- [9] Yoonchang Han, Jaehun Kim, Kyogu Lee, Yoonchang Han, Jaehun Kim, and Kyogu Lee. Deep convolutional neural networks for predominant instrument recognition in polyphonic music. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 25(1):208–221, 2017.
- [10] Philippe Hamel, Sean Wood, and Douglas Eck. Automatic Identification of Instrument Classes in Polyphonic and Poly-Instrument Audio. *Ismir*, pages 399–404, 2009.
- [11] Jordi Pons, Olga Slizovskaia, Rong Gong, Emilia Gómez, and Xavier Serra. Timbre Analysis of Music Audio Signals with Convolutional Neural Networks. 2017.
- [12] Philippe Hamel and Douglas Eck. Learning Features from Music Audio with Deep Belief Networks. *International Society for Music Information Retrieval Conference (ISMIR)*, (Ismir):339–344, 2010.
- [13] Honglak Lee, Pt Pham, Y LARGMAN, and Ay Ng. Unsupervised feature learning for audio classification using convolutional deep belief networks. *Nips*, pages 1–9, 2009.
- [14] Eric J Humphrey, Aron P Glennon, and Juan Pablo Bello. Non-linear semantic embedding for organizing large instrument sample libraries. In *Machine Learning and Applications and Workshops (ICMLA), 2011 10th International Conference on*, volume 2, pages 142–147. IEEE, 2011.
- [15] John Thickstun, Zaid Harchaoui, and Sham Kakade. Learning Features of Music from Scratch. pages 1–14, 2016.
- [16] Soroush Mehri, Kundan Kumar, Ishaan Gulrajani, Rithesh Kumar, Shubham Jain, Jose Sotelo, Aaron Courville, and Yoshua Bengio. SampleRNN: An Unconditional End-to-End Neural Audio Generation Model. pages 1–11, 2016.
- [17] Jesse Engel, Cinjon Resnick, Adam Roberts, Sander Dieleman, Douglas Eck, Karen Simonyan, and Mohammad Norouzi. Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders. 2017.
- [18] Andy M Sarroff and Michael Casey. Musical Audio Synthesis Using Autoencoding Neural Nets. *Proceedings of the International Computer Music Conference*, 1(September):14–20, 2014.
- [19] Shinji Takaki and Junichi Yamagishi. A deep auto-encoder based low-dimensional feature extraction from fft spectral envelopes for statistical parametric speech synthesis. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pages 5535–5539. IEEE, 2016.
- [20] Spyridon Stasis, Ryan Stables, and Jason Hockman. A Model for Adaptive Reduced-Dimensionality Equalisation. *Proc. of the 18th Int. Conference on Digital Audio Effects (DAFx-15)*, pages 1–6, 2015.
- [21] Takaaki Ishii, Hiroki Komiyama, Takahiro Shinozaki, Yasuo Horiuchi, and Shingo Kuroiwa. Reverberant speech recognition based on denoising autoencoder. In *InterSpeech*, pages 3512–3516, 2013.
- [22] Emmanouil Benetos, Margarita Kotti, and Constantine Kotropoulos. Musical in-

- strument classification using non-negative matrix factorization algorithms and subset feature selection. In *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, volume 5, pages V–V. IEEE, 2006.
- [23] Pablo Sprechmann, AM Bronstein, and Guillermo Sapiro. Supervised non-negative matrix factorization for audio source separation. *Vista.Eng.Tau.Ac.II*, pages 1–14, 2014.
- [24] G. E. Hinton. Reducing the Dimensionality of Data with Neural Networks. *Science*, 313(5786):504–507, 2006.
- [25] Matthias Scholz, Martin Fraunholz, and Joachim Selbig. Nonlinear principal component analysis: neural network models and applications. *Principal manifolds for data visualization and Dimension Reduction*, pages 45–68, 2008.
- [26] Pascal Vincent PASCALVINCENT and Hugo Larochelle LAROCHEH. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion Pierre-Antoine Manzagol. *Journal of Machine Learning Research*, 11:3371–3408, 2010.
- [27] Daniel Griffin and Jae Lim. Signal estimation from modified short-time fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(2):236–243, 1984.
- [28] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. pages 1–15, 2014.
- [29] Gérard Grisey. *Partiels [: pour 18 musiciens: partition]*. Ricordi, 1976.

# Vivace: a collaborative live coding language and platform

Vilson Vieira<sup>1</sup>, Guilherme Lunhani,  
Geraldo Magela de Castro Rocha Junior, Caleb Mascarenhas Luporini,  
Daniel Penalva, Ricardo Fabbri<sup>2</sup>, Renato Fabbri<sup>3</sup>

<sup>1</sup>Cod.ai  
SC, Brazil

<sup>2</sup>Polytechnic Institute – IPRJ/UERJ  
Rua Bonfim, 25 - Vila Amélia – CEP 28625-570, Nova Friburgo, RJ, Brazil

<sup>3</sup>Visualization, Imaging and Computer Graphics lab – VICG/ICMC/USP  
Av. Trabalhador São-Carlense, 400 - Centro – CEP 13566-590, São Carlos, SP, Brazil

`vilson@void.cc`, `rfabbri@gmail.com`, `renato.fabbri@gmail.com`

## Abstract

Live coding is a performance and creative technique based on improvised and interactive coding. Many recent endeavors have focused in live coding both because of aesthetics and as a way to alleviate performance drawbacks when the musical instrument is a computer. This paper describes the principles and the design of Vivace, a live coding language and environment built with Web technologies to be executed on web browsers. The approach is compelling by 1) allowing many performers to code simultaneously; 2) the synthesis of audio and video; 3) a very simple syntax; 4) being a multiplatform software. We also strive to contextualize Vivace by means of historical and usage summaries including a live coding sub-genre.

## 1. Introduction and narrative

Live coding is an artistic performance and creative technique based upon writing software code in a live and improvised manner [1]. It can be used to generate e.g. sound, images, video, lights and poetry although it is prevalent in computer music [2]. Most often, the code is continually changed and projected in a large surface as a way to make it visible to the audience [3]. The usage of live coding in non-performative contexts is also reported, such as in sound design and art installations [2]. In this paper, we describe Vivace, a live coding language and platform. It emerged from pragmatic and aesthetic needs, as described in the following sections. The software

is cross-platform because it is based on web technologies, such as HTML5 Video and Web Audio API, and is oriented towards video and music rendering. The language is very simple, allowing for a primary goal of Vivace to be achieved: the simultaneous writing of the code by many performers and the audience.

In summary, this paper describes how a number of artistic presentations motivated the creation of the language and platform, describes the Vivace language and platform and how this endeavor lead to the emergence of a live coding sub-genre called “freak coding” (e.g. by its manifesto [4] and related artists).

### 1.1. A historical outline

In November of 2011, a live coding trio called *FooBarBaz* [5] unleashed its first presentation for a wide audience. Its performers used two instances of ChucK [6] and a dedicated mixing instance composed by Puredata and an analog mixer <sup>1</sup>. Live coding had been gaining world wide popularity [1, 7, 8, 3] which motivated the creation of a dedicated congress, the International Conference on Live Coding (ICLC) [9], already on its third edition, and a network of public events and movements like Algorave [10]. Live Coding has been adopted by performers also in Brazil, like Guilherme Lunhani, Antonio Goulart, André Damião Bandeira and Magno

<sup>1</sup>Pictures of the presentation available at <http://www.flickr.com/photos/festivalcontato/6436260557>

Caliman. However, when considering massive audience, live coding practice remains quite untouched in Brazil. To the best of our knowledge, the presentation was the first live coding performance in our country with such a wide audience – almost 5,000 attendees were in the gathering where code was used on-the-fly to create the music they were listening. At the same time, two live coding desktop work-spaces were projected on large screens to the public, following the principles of the TOPLAP manifesto [11].

During the performance, the trio used ChuckK in an unconventional way. Instead of writing loops and conditionals, one of the live coders manipulated parameters of audio files by editing lists of numerical values together with mnemonic operations like retrograde and transposition. The other live coder focused on more fluid lines with large sounds with evolving characteristics; this contributed for coherent musical arcs. Audio mixing with Puredata was carried out by the third performer literally using handwaving gestures tracked by a camera and custom color detection algorithms designed by us. Live coders used code templates for quick insertion in the text editors (Vi and Emacs). Other visual resources the performers focused on: Unix “cowsay” generated phrases and animated bouncing balls – stimulating the audience to imitate Rapid Eyes Movements (REM) – on both terminals, i.e. on both screens projected to the audience. The performance was reported as interesting by technicians, artists and the general audience. Nonetheless, it was altogether complex, not to say messy.

Based on the aforementioned elements, and the need for greater simplicity and interactivity with the public, the Vivace was designed as a new live coding language and platform <sup>2</sup> [12]. To avoid software configuration, and to make it easy to share the session and the system, the Web was chosen as the running environment for Vivace. On every new session performed using Vivace, new principles were added into the language and, at the same time, into our artistic approach.

<sup>2</sup>Live demonstrations of Vivace are on-line at <http://void.cc/freakcoding> and <http://void.cc/cranio>, ready to be used by everyone using Google Chrome, Mozilla Firefox or Apple Safari.

## 1.2. Additional motivation & inspiration: arrange the room, the code is dirty

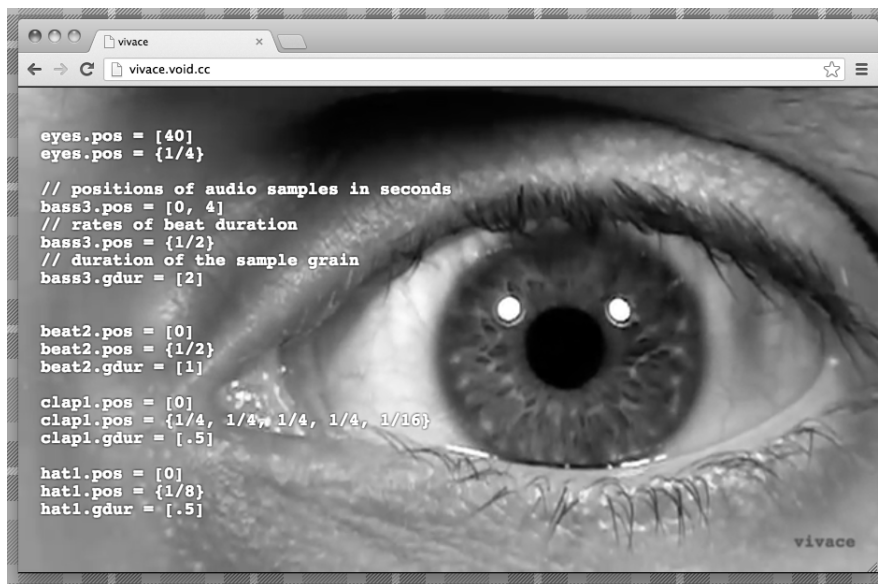
Vivace is inspired by various live coding languages. The syntax of Vivace, as shown in Figure 1, borrows elements from *ixi lang* [13] such as the use of sequences to control audio parameters in real time. ABT [14] and FIGGUS [15] were tightly relevant to the development of Vivace as well and we are planning to rewrite some of their components – originally in Python – inside Vivace. ChuckK was an influence from the beginning, and Vivace resembles the simplified interfaces that we constructed in ChuckK which were often minimized into lists in few lines. Fluxus [16] was also inspirational for the Vivace environment where the code is shown on top of the video frames.

After the creation of the Audio Data API [17] and the most recent Web Audio API [18] – easing real time audio processing inside Web browsers – a collection of audio Web applications started to emerge. The same holds for live coding languages and systems. Thus, in addition to the *desktop languages* above, Vivace was also inspired by recent Web live coding languages and is part of this *family* of Web applications together with Gibber [19], livecoder [20], livecoding.io [21], livecodelab [22] and Wavepot [23] to mention just a few.

A remarkable difference between Vivace and other languages and environments in the same family is the element of collaborativity. Vivace was built to enable writing code by many hands at the same time, as with the now popular “e-pads” or collaborative real-time text editors <sup>3</sup>, a feature which is naturally implementable on the Web. Another difference is the unconcern to be a Turing-complete language. This made the design of Vivace more flexible and closer to musical thinking as opposed to a computing process (a characteristic perceived in *ixi lang* as well). Vivace is designed to associate the precision of code and the flexibility of artistic expression while maintaining simplicity.

<sup>3</sup>Etherpad on: <http://etherpad.org>.





**Figure 1: The Vivace platform basic interface: video playback is in the background while the code is in the foreground and controls both video and musical audio. The interface is accessed as a usual HTML page in a web browser.**

## 2. The language (specification) we all speak

Vivace, as a language <sup>4</sup>, is a collaborative live coding language with use of extremely simple syntax, mnemonic operations, easy audio mixing, template editing and audio parameters automation. The use of shared code, sounds and images leads to a more complex scenario, thus increasing the possibility of inconsistency of compiled code as well as artistic results.

Vivace is not an imperative language. Instead of routines and procedures to control audio attributes, it uses definitions related with musical scores and the *track paradigm* common on music production software [3]. It is natural to musicians (and, as we experienced during performances, also to non-musicians) to understand a sequence of notes, or audio parameters, repeating over and over again, than for-loops and if-chains. In this way, Vivace is a declarative, domain specific language, based on the following principles:

<sup>4</sup>The complete specification can be found at <https://github.com/automata/vivace/wiki/Language-spec-together-with-the-grammar> (<https://github.com/automata/vivace/blob/master/vivace.json>) and lexical rules (<https://github.com/automata/vivace/blob/master/vivace.jsonlex>) specified in Bison and Flex dialects, respectively.

- Names are literals like *foo*, *bar*, *baz* and are defined as the user wants.
- Music is constituted by voices (instruments).
- Voices have name, timbre and parameters changing along time.
- The language should be simple. One only defines some properties with a set of values (i.e. arrays, dictionaries) making it possible to generate sequences.
- Mnemonic musical operations (reverse, inverse, transpose) on properties by use of syntax sugar: few chars, powerful changes.
- Timbre are signals made by chains of audio generators and filters or video files as described below.
- Parameters are musical notes, amplitudes, oscillator frequencies, delay time and so on.
- Parameters change their values at specific times and for certain durations.

Here is a “Hello, World!” Vivace code <sup>5</sup>:

```

# foo is a simple audio sample,
# oscillator or video file
foo.src = youtube('YOUTUBE_URL')
# defining the video positions
  
```

<sup>5</sup>YOUTUBE.URL stands for any youtube video url such as <http://www.youtube.com/watch?v=XXX>

```
# (in seconds) to be played
foo.pos = [10, 20, 35]
# the durations,
# as time ratios of a pulse,
# to be played at each position:
foo.cdur = [1/2, 1/4, 1/8, 1/16, 1]
```

A voice is defined as *foo* and its parameters are specified using the *dot* operator. Every parameter changes over time as the values written in numerical sequences, surrounded by brackets. A special sequence exists to every parameter. This is essentially all of the Vivace syntax.

There are extra semantics to operate on the sequences. Every sequence accepts operators: mnemonic commands used to reverse, transpose and even replace elements of the sequence based on list comprehensions. Those operations are common in music composition [3] and having them as mnemonics makes typing fast and handy for live coding. The next listing presents the standard operators:

```
# one can use operators
foo.pos = [1, 2, 3] reverse
# result is [3, 2, 1]
foo.pos = [1, 2, 3] inverse
# result is [1, 0, -1]
foo.pos = [1, 2, 3] transpose +2
# result is [3, 4, 5]

# list comprehension
foo.pos = [i+1 for i in [1, 2, 3]]
# result is [2, 3, 4]

# or combine both
foo.pos=[i+1 for i in [1, 2]] reverse
# result is [3, 2] as expected
```

Vivace is written in JavaScript to take advantage of Web technologies. To parse Vivace, Jison [24] comes handy, a JavaScript library that clones Flex and Bison functionality as lexer and parser. This flexibility to parse and execute new languages as JavaScript inside every browser opens a remarkable opportunity to experiment with new syntax and semantics for live coding. To make the Vivace editor collaborative we used ShareJS [25] which makes Web applications content live concurrent. ShareJS uses a multiple clients and one server network architecture. By running a common server for both Vivace Web application and files, it is possible to share Vivace code with any client accessing a common URL. Furthermore, considering the tradition of UI design and development on the Web thanks to HTML and CSS, one can experiment

those new languages with fast prototyped UI – a requisite already addressed by live coding languages [26, 27] like Texture [28], Al-Jazari and Betablocker. Along these advantages, it is important to note: every live coding language built on the Web runs everywhere a browser is installed. No firewall chain to bypass for OSC, no software installation and configuration, no dependencies, people just need to type an URL.

## 2.1. Vivace audio and video engine

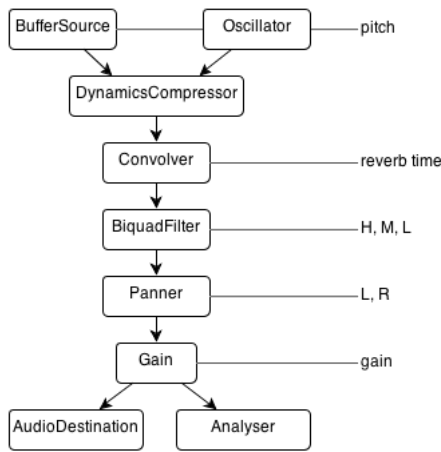
Before the Web Audio API, the only way to create sound in web pages was using plug-ins. Recently, the Web Audio API enabled real time audio processing on Web browsers<sup>6</sup>. Every functionality is implemented as native code (in C++ and Assembly when appropriate) to guarantee maximum performance. The API is based on a convenient and familiar paradigm: audio unit graphs. Web Audio specifies a collection of nodes (*AudioNode* objects) and routines to connect and disconnect them. While manipulating those nodes we can create a large number of audio applications: synthesizers, filters, analyzers, mixers and even real time audio engines for live coding. This motivated basic explorations of multichannel expansion, filtering and audio effects, controlling an integrated Web audio system.

Every voice in Vivace is represented as a default audio chain such as the one shown in Figure 2. All audio unit parameters within this chain (e.g. pitch, reverb time, high, medium and low channel levels, panner values and gain) can be manipulated editing the code or by sliders on a GUI (Figure 3).

This kind of interface is more familiar to musicians, resembling a real mixer, and enables an adequate treatment of voice timbre and spatialization of the sound sources by means of parameters like level of stereophonic channels L and R, quality, central frequency and gain of a 3-band equalization filter, and reverb time control.

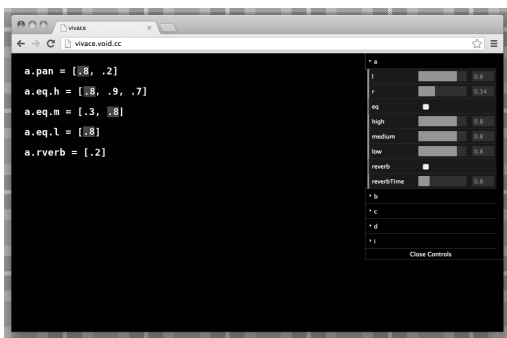
Vivace supports every audio unit implemented

<sup>6</sup>At the time this paper was written, more than 74% of current available Web browsers support Web Audio API [29], including the most populars Google Chrome, Mozilla Firefox and Apple Safari.



**Figure 2: Standard audio processing units are directly related to Web Audio API objects for each voice within Vivace.**

by Web Audio API. It is possible to load audio files or synthesize in real time using wave-table oscillators. The default audio chain of each voice can be modified at any time while it is running. It is interesting to note the presence of an “Analyzer” inside the default chain. It uses FFT (natively implemented) to expose energies and frequencies, enabling the use of those values to animate videos and render graphical forms inside Vivace.



**Figure 3: Every audio unit parameter can be manipulated by code or using the UI in the Vivace platform.**

Along with audio, Vivace supports video files. It is possible to upload files or use YouTube URLs. Videos are treated the same way as buffer sources or oscillators, i.e. as voices, and can be manipulated in real time, making Vivace a live cinema or a VJing tool.

### 3. Into the wild: the rise of freak coding makes it collaborative

Vivace as a tool enables interaction while everyone can use their own creativity. The interaction is not mediated by a common score, but by a mutual desire to create a composition in real time. In this context the *freak coder* was born (Figure 4): someone that adds his individuality with others, aiming to transform the computer into an instrument of artistic fruition, without restricting to himself the control of the machine but inviting everyone to join him in the activity. A freak coder decides what he is going to do and amplifies his own comprehension of the computer capacity as an instrument. By using simple rules, Vivace enables the emergence of the performance and makes it a kind of a collective game, where the rules, being visible to everyone through the code, eases audience and specialists alike to join in.



**Figure 4: Freak coder: a live coder who uses popular and “freak” media to boost the attention of the audience to the shared code.**

Live coding becomes a natural path to the type of use and technological development in which freak coders are involved, in confluence with the understanding that technology should never be treated as a dogma or kept in secret. Live coding is seen as a behavioral de-alienation of a digital artist. The triad performer, code and audience characterizes the performance as live coding. This comprehension was possible after a presentation by labMacambira.sourceforge.net at the 9<sup>th</sup> edition of AVAV (*AudioVisual Ao Vivo* or Live Audiovisual), an event where artists who are experimenting with audio and video in real time come together to show their works. In this presentation, the authors Caleb Luporini and Gera Rocha started without Renato Fabbri and Vilson

Vieira, as they were on their way to the presentation, traveling from another city. Upon arrival, Fabbri and Wilson turned their laptops on and started taking part on the performance in such a way that no embarrassment or rupture was brought into the event. It is important to state that no previous rehearsal had taken place between Mr. Luporini and Mr. Rocha.<sup>7</sup> In the 30 minutes-long performance, the audience started to take guidance from messages given on the performance large screen and actually edited Vivace code that was being played together with the starting four performers.

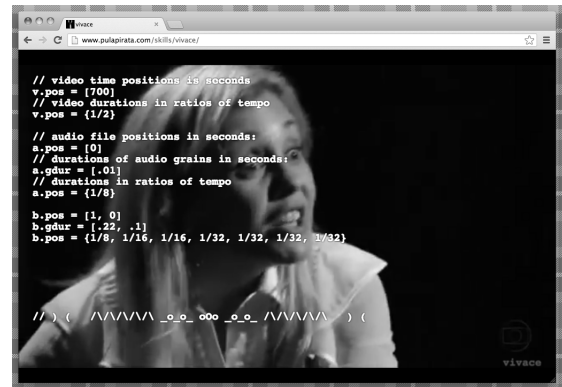
Another artifact noted on the presentation was the emergence, in a formal environment<sup>8</sup>, of a collective euphoria fertilized by a human-machine interaction. It is the performer's posture that takes a spectator to an experience of a non-spectator and to take part on a highly technological activity as something playful and possible to be assimilated. During the entire presentation, all labMacambira.sourceforge.net members were cheerful and established a relation of lightness and brotherhood with the audience. Spectators were being constantly invited by the posture of labMacambira.sourceforge.net members to interact with what was being proposed. This interplay between the four elements therein present – performers, computer, Vivace and audience – created an environment of collaboration and liberty as generators of playfulness and technical knowledge unheard of, at least in Brazilian live coding, to our knowledge. This is the “facilitator” that emerged and received the name *freak coder*.

To attract the attention of the wider audience, we as freak coders used popular media as material. The code was displayed in front of video scenes sampled from popular Brazilian novels (as in Figure 5) and B-movies, which resulted in a “freak” style, with images of monsters and funny dialogues between novel actors. In other performances for heterogeneous attendees the effect was the same as the first presentation where we used these kinds of pop-art: the people was

<sup>7</sup>Videos were selected beforehand by Mr. Luporini alone without knowledge of the other performers.

<sup>8</sup>The four performers were in a light-less room, three of them facing the big screen and the other one facing the public.

fascinated by the adherence between the code and the media they see every day on their TV sets. Since then, the use of popular and “freak” media has become a signature of “freak coding”.



**Figure 5: Videos sampled from popular Brazilian novels and B-movies were used as material to attract the attention of the audience to the code in the “freak coding” aesthetics and live coding sub-genre.**

All labMacambira.sourceforge.net members take part in the Brazilian free software movement. In a way, freak coding origins should be looked for inside this movement. It is inherent to the free software movement the continued transmission of what is known. The same happens on the demystification of technology and the festive and gregarious behavior. At the performer-computer relation is where this behavior becomes concrete. More than the materials used in the live coding sessions, the performer's stance in relation to the computer – as already expressed in the described presentation – is what really subverts not just the highly technical computer use but the relationship between the human and the machine. Namely a kind of a “rock and roll” stance. The freak coder breaks, by his own nature, the stigma of the computer as the source of a serious and professional posture. In the same way, breaks with the posture of the scholar performer, stern and closed in herself. The freak coding is “rock and roll”. The freak coder becomes Jerry Lee of technology making “technopyrophagy”. He codes and cheers at same time. The freak coder seduces through the computer screen and by the way he codes.

#### 4. Conclusions and future work

Vivace was motivated by actual performances that took place in the recently emerging Brazilian live coding scene. The development of the language was guided by this direct contact of performers and the wider public. The language was designed and implemented after the identification of common patterns already used on presentations and the need for simplicity and interactivity. Following open source practice, Vivace is developed by many hands from computer scientists, musicians, activists and social scientists. At present, the language is certainly not perfect nor all-encompassing, but it does strike a useful balance between flexibility and rigor, making it an interesting language for artistic expression on collaborative sessions.

It is important to note the advantage of using the Web as the platform for experimentation on live coding and other computer music approaches. Recent APIs like Web Audio together with the rapid prototyping of multi-platform UI and language parsers creates a prolific scenario. Henceforth the most interesting characteristic is the collaboration proportioned by the Web. Using collaborative editors we can expose an entire music program to be edited by anyone, anywhere.

Vivace, although a “freak coding” language, is constrained in its music expression. Having a domain specific language as Vivace is interesting to express some musical ideas where it is hard or even impossible elsewhere. In this context we assume Vivace as one of the many tools and a contribution to create other languages and collaborative systems emerging from live coding practices. In this way, we can tell that the described performances and even Vivace are motivating the creation of other live coding tools. Carnaval<sup>9</sup> is one of these new realizations, it can be seen as a “personal TV channel”. Each channel is related to a Vivace instance, making it possible for anyone to remix media and create their own composition. It is a social network of live coded remixes. Vivace, instead of an isolated piece of

<sup>9</sup>Carnaval is being conceived as free software and a collaborative art piece since its beginning. The first sketches are on-line at <http://automata.cc/carnaval>

software is then used as a module, a part of Carnaval.

In our experiences as performers and developers of live coding languages we can assert this style of music realization as inspirational and flexible. Nevertheless, we continue to search for improvements on Vivace – and others derived tools – to increase an already consolidated objective of live coding as a musical practice: make computer music performance more human, more interactive with the wider audience [3].

Future improvements are planned on Vivace: the possibility to explicitly define large musical arcs as nested sequences related to audio units, the use of 3D graphics APIs to render forms – and relate them to running audio parameters – and text messages to the audience, and improved UI to make the code editing more flexible and reactive [30]. Along with the language and system itself, this paper is a live initiative. Freak coding as an artistic style (a sub-genre of live coding) will be explored more deeply on future studies – regarding its aesthetics – and in already planned performances.

We want to end by underlining the importance of social aspects regarding live coding. The authors were not working physically close to each other since the beginning, i.e. since the creation of Vivace and the rise of freak coding. The performances emerged from the gathering of the artists and programmers, which gave rise to new tools and aesthetics. On the other hand, given the continuous audiovisual feedback the audience and performers have from the code, we noticed that the potential of live coding for computer programming demystification and introduction is often emphasized by enthusiasts and the literature [2].

#### References

- [1] C. Nilson. Live coding practice. *Proceedings of New Interfaces for Musical Expression (NIME)*, 2007.
- [2] Marc J Rubin. The effectiveness of live-coding to teach introductory programming. In *Proceeding of the 44th ACM technical*

- symposium on Computer science education*, pages 651–656. ACM, 2013.
- [3] N. Collins. Live coding of consequence. *Leonardo*, 44(3):207–211, 2011.
- [4] Guilherme Lunhani, Geraldo Magela, Vilson Vieira, Caleb Luporini, and Renato Fabbri. Freak coding manifesto. <http://pontaopad.me/freakcoding>, 2012.
- [5] Ricardo Fabbri Renato Fabbri, Vilson Vieira. Foobarbaz: Metasyntactic variables. <http://wiki.nosdigitais.teia.org.br/FooBarBaz>, 2011.
- [6] G. Wang, P.R. Cook, et al. Chuck: A concurrent, on-the-fly audio programming language. In *Proceedings of the International Computer Music Conference*, pages 219–226. Singapore: International Computer Music Association (ICMA), 2003.
- [7] N. Collins, A. McLean, J. Rohrhuber, and A. Ward. Live coding in laptop performance. *Organised Sound*, 8(03):321–330, 2003.
- [8] A.R. Brown and A.C. Sorensen. aa-cell in practice: An approach to musical live coding. In *Proceedings of the International Computer Music Conference*, pages 292–299. International Computer Music Association, 2007.
- [9] Live Code Research Network. International conference on live coding. <http://iclc.livecodenetwork.org/>, 2017.
- [10] Algorave. Algorave. <http://algorave.com>, 2017.
- [11] A. Ward, J. Rohrhuber, F. Olofsson, A. McLean, D. Griffiths, N. Collins, and A. Alexander. Live algorithm programming and a temporary organisation for its promotion. In *Proceedings of the README Software Art Conference*, 2004.
- [12] Vilson Vieira and Renato Fabbri. Vivace. <http://automata.github.com/vivace>, 2012.
- [13] T. Magnusson. ixi lang: a supercollider parasite for live coding. In *Proceedings of the International Computer Music Conference*. University of Huddersfield, 2011.
- [14] Renato Fabbri. A beat tracker. <https://sourceforge.net/p/labmacambira/abt/>, 2008.
- [15] Renato Fabbri. Figgus: Finite groups in granular and unit synthesis. <http://wiki.nosdigitais.teia.org.br/FIGGUS>, 2012.
- [16] David Griffiths. (fluxus). <http://www.pawfal.org/fluxus/>, 2013.
- [17] David Humphrey, Corban Brook, Al MacDonald, Yury Delendik, Ricard Marxer, and Charles Cliffe. Audio data api. [https://wiki.mozilla.org/Audio\\_Data\\_API](https://wiki.mozilla.org/Audio_Data_API), 2010.
- [18] Chris Rogers. Web audio api: W3c working draft. <http://www.w3.org/TR/webaudio/>, 2012.
- [19] Charlie Roberts. Gibber. <http://gibber.mat.ucsb.edu/>, 2012.
- [20] Fritz Obermeyer. Livecoder. <http://livecoder.net>, 2012.
- [21] Gabriel Florit. livecoding.io. <http://livecoding.io>, 2012.
- [22] Davide Della Casa and Guy John. Live-codelab. <http://livecodelab.net>, 2012.
- [23] George Stagas. Wavepot. <http://wavepot.com>, 2015.
- [24] Zach Carter. Jison. <http://jison.org>, 2010.
- [25] Joseph Gentle. Sharejs. <http://www.sharejs.org>, 2011.
- [26] A. McLean, D. Griffiths, N. Collins, and G. Wiggins. Visualisation of live code. *Proceedings of Electronic Visualisation and the Arts 2010*, 2010.
- [27] T. Magnusson. Algorithms as scores: Coding live music. *Leonardo Music Journal*, pages 19–23, 2011.
- [28] A. McLean and G. Wiggins. Texture: Visual notation for live coding of pattern. In *Proceedings of the International Computer Music Conference*, 2011.
- [29] Alexis Deveria. Can i use web audio api. <http://caniuse.com/audio-api/embed/>, 2017.
- [30] Bret Victor. Learnable programming: Designing a programming system for understanding programs. <http://worrydream.com/LearnableProgramming/>, 2012.

# Web Audio application development with *Mosaicode*

Flávio Luiz Schiavoni<sup>1</sup>, Luan Luiz Gonçalves<sup>1\*</sup>, André Lucas Nascimento Gomes<sup>1</sup>

<sup>1</sup>Universidade Federal de São João Del Rei – UFSJ  
Computer Science Department - DCOMP  
São João Del Rei, MG – Brazil

fls@ufsj.edu.br, luanlg.cco@gmail.com, andgomes95@gmail.com

## Abstract

The development of audio application demands a high knowledge about this application domain, traditional programming logic and programming language. It is possible to use a Visual Programming Language to ease the application development, including experimentations and creative exploration of the Language. In this paper we present a Visual Programming Environment to create Web Audio applications called *Mosaicode*. Different from other audio creation platforms that use visual approach, our environment is a source code generator based on code snippets to create complete applications.

## 1. Introduction

Since the emergence of Web Audio, the web browser has been used as a platform to create and process real time audio. This API leaded several developers and artists to explore and use the possibility of creating sounds and music in a portable format that can reach different operating systems, architectures and devices, from desktops to mobiles. In the raise of the Internet, the web was used only to distribute music and the sound interaction was possible only based on recorded audio in digital formats. Nowadays, it is possible to reach a new level of interaction and use the browser as a powerful tool for sonic interaction.

Web technologies provide powerful tools to develop cross-device applications including a high level sample-accurate sound engine and a sophisticated layout system for visual feedback [1]. It also enables to incorporate accelerometers, multi touch screens, gyroscopes

and other sensors available on mobile devices [2, 3, 4, 5]. It is also possible to incorporate legacy music interfaces with the Web MIDI API [6]. Nonetheless, digital artists often have difficulty starting their research and working with digital art due to lack of knowledge of algorithms and traditional programming logic.

To simplify the development of applications, it is possible to use Visual Programming Languages (VPLs). VPL is a class of programming language that allows the programmer to develop software using a two-dimensional notation and interact with the code by the means of a graphical representation instead of editing an one-dimensional stream of characters, having to memorize commands and textual syntaxes [7]. This may allow non-programmers or novice programmers to develop complete applications [8] since VPLs can bring ease to system development. In addition, code abstraction by means of a diagram can bring practicality in changing the code making them quite suitable for rapid prototyping [9]. This is a known approach in arts due to the common use of tools like Pure Data, Max/MSP or Eyesweb.

Another way to simplify the development of computational systems is by using domain-specific languages (DSL) [10]. DSLs have the knowledge of the domain embedded in their structure and are at a higher abstraction level than general-purpose programming languages. It makes the process of developing systems within your domain easier and more efficient because DSLs require more knowledge about the domain than programming skills [11]. For this reason, the potential advantages of DSLs include reduced maintenance costs through re-use of built-in features and increased portability, reliability,

\*Supported by UFSJ.

optimization and testability [12]. Domain Specific Languages are also common in art field since this approach was used to develop languages like CSound, Supercollider or RTCMix.

In this paper we present *Mosaicode*, a visual programming environment that can be used to develop systems in the specific domain of digital art combining the simplicity of visual programming with code reuse of DSLs. We propose in this work, the construction of a set of Blocks for audio application based on *JavaScript* programming language and the Web Audio API.

This work is organized as follows: Section 2 presents related works, Section 3 presents the *Mosaicode* application, Section 4 presents the development of a Block set to *Mosaicode* and JavaScript/Web Audio, Section 5 presents initial discussions of our research. Finally, Section 6 presents Conclusion and Future Works.

## 2. Related Works

From the point of view of VPLs to audio and music programming, our related works start with Pure Data, Max/MSP and Eyesweb. Pure Data<sup>1</sup> is a Visual Programming Environment for Sound and Music that plays host to GEM environment[13] to 3D graphic processing [14]. Max/MSP<sup>2</sup> is also a music and video real time graphical programming environment [15]. Pure Data and Max/MSP share the same paradigm being both created by the same author, Miller Puckete. Eyesweb<sup>3</sup> is a project focused on real time analysis of body movement and gesture[16].

From the point of view of JavaScript and Web Audio programming into an easier form of programming, related works are several programming libraries like Flocking.js[17], gibber[18], WebCsound[19] and others.

From the point of view of Code generators, Processing<sup>4</sup> is a DSL textual programming language and an IDE that generates code to Graphi-

<sup>1</sup>Project Website: <http://puredata.info>.

<sup>2</sup>Project Website: <https://cyclimg74.com/products/max>.

<sup>3</sup>Project website: [http://www.infomus.org/eyesweb\\_ita.php](http://www.infomus.org/eyesweb_ita.php).

<sup>4</sup>Project Website: <https://processing.org/>.

cal Art development. Beyond Processing, a sister project called Processing.js [20] was designed to write visualizations, images, and interactive content. There is also p5.js a JavaScript library based on the core principles of Processing.

Another interesting code generator is Faust[21], a purely functional programming language to signal processing that is able to generate code to several target languages, libraries and APIs.

The idea of using a VPL to generate web based music application was already explored by EarSketch [22] but in this case, a web based environment was used to create the applications based on the Blockly programming environment.

## 3. About the *Mosaicode*

Initially developed as a Computer Vision Programming Environment to generate C code to OpenCV library, *Mosaicode* has been expanded to other programming languages and domains.

This tool is a visual programming environment that uses the **Block** metaphor to create computer programs. Blocks are organized into groups in our environment GUI, presented on Figure 1. A Block is the minimal source code part and brings the abstraction of a functionality of our desired domain. Blocks have static properties, used to set up their functionality, presented in Figure 2.

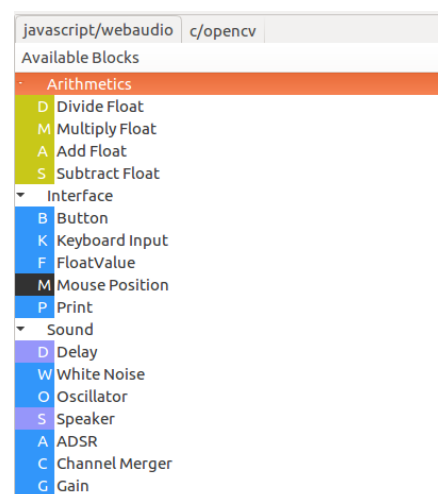


Figure 1: Part of Javascript/Web Audio Block groups



Blocks also have dynamic properties whose values can be set up by other Blocks. This capability to exchange information is represented by the Blocks input/output **Ports**. The information exchange by different Blocks is made creating a **Connection** between two or more Ports. A Block Port has a defined type and a Connection can be done using ports of the same type.

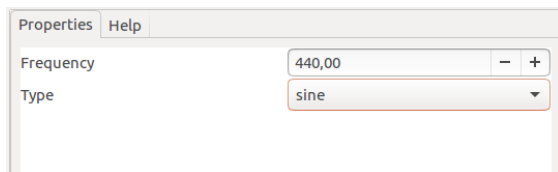


Figure 2: Oscillator Block static properties

The Collection of Blocks and Connections creates a **Diagram**, as presented in Figure 3.

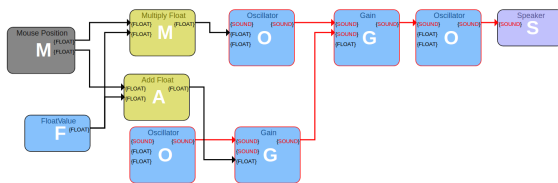


Figure 3: A Diagram with several Blocks interconnected.

Despite it can be seen as a Visual Programming Language, like Pure Data or Max/MSP, *Mosaiccode* is not an interpreted environment but a code generator. Every single Block and Connection define code fragments and add code Snippets to the application final code. The application code also depends on a Code template that defines how the Code Snippets will be merged into the final application.

Once the Diagram is completed, it is possible to generate the source code and to run it. Figure 4 presents a generated source code.

The tool has an interface and a plugin manager that allows the creation of new components for the environment, so the tool can be extended, allowing the generation of source code to different programming languages and specific domains.

```

17 var block_20 = context.createOscillator();
18 var block_20_o0 = null;
19 var block_20_i1 = function(value){
20   block_20.frequency.value = value;
21 };
22 var block_20_i2 = function(value){
23   oscillator = ''
24   if (value < 1) oscillator = 'square';
25   if (value == 1) oscillator = 'sine';
26   if (value == 2) oscillator = 'sawtooth';
27   if (value > 2) oscillator = 'triangle';
28   block_20.type = oscillator;
29 };
30
31 // block_60 = Mouse
32 var block_60_o0 = [];
33 var block_60_o1 = [];
34
35 // block_66 = Multiply Float
36 var block_66_arg1 = 0;
37 var block_66_arg2 = 0;
38 var block_66_o0 = [];
39

```

Figure 4: Example of generated Source Code.

## 4. Developing Web Audio Blocks to the *Mosaiccode*

Once we started developing a Block set to JavaScript/Web Audio code generation, we investigated which Blocks could be necessary on the environment to satisfy the Digital Arts requirements. The Blocks creation was based on other tools like Gibberish and Pure Data.

We investigated Gibberish and this library has a collection of Audio processing classes classified into the following categories: Oscillators, Effects, Filters, Synths, Maths and Misc. The Math group contains Add, Subtract, Multiply, Divide, Absolute Value, Square Root and Pow [2]. The Misc group contains Sampler objects (play/record), Envelope (ADSR, AD), Line Ramp and others. Later, authors added to this list a Drums category[3] and GUI widgets like Sliders, Buttons, Sensors, knob, Piano and other GUI components.

We also investigated Pure Data native objects and this tool has an interesting object list organized into categories: General, Time, Math, MIDI and OSC, Misc, Audio Math, General Audio Manipulation, Audio Oscillators And Tables and Audio Filters.

Our last investigation included an analysis of Web audio Native Nodes and the possibilities that our target language and API could offer.

### Native Nodes

Our first set of Blocks was done using the Web Audio Native Nodes. It included Audio Oscillators, Audio Filters, Audio Effects and General Audio Manipulation like Gain, Speakers, Channel Merge and Channel Splitter.

These Nodes have two kind of configuration parameters: Fixed parameters and a-rate Audio Parameter. Fixed parameters, like `OscillatorNode.type`, has fixed values like “sine”, “square”, “sawtooth”, “triangle” and “custom”. This parameters were directly mapped to a Block static property in *Mosaicode*. The other parameter type, a-rate Audio Parameter, like `OscillatorNode.frequency`, were mapped as an input port. This kind of parameter has also a value field (`OscillatorNode.frequency.value`) also mapped to a static property.

The `AudioNode` channel inputs were mapped to input ports and channel outputs were mapped to output ports. These ports are connectible and follow the basic idea of Node connection from Web Audio API, presented on Figure 5.

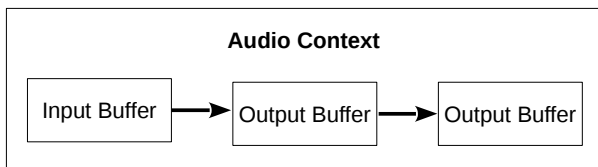


Figure 5: A typical workflow for Web Audio Nodes [23].

### Script Processor Node

The high-level Nodes provided by the Web Audio API do not include all possibilities and necessities found in our investigation. To attend other sound processing features, we used a special node called `ScriptProcessorNode` that enables users to define complete audio systems entirely in JavaScript [1].

The Script processor Node, presented in Figure 6, allows one to develop new Sound Nodes and integrate it to the Web Audio API.

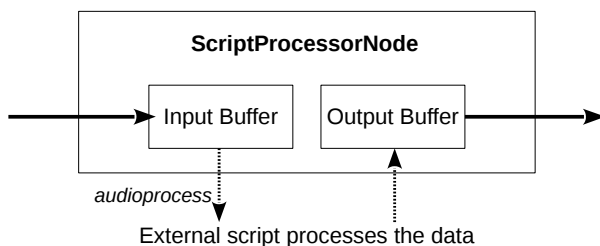


Figure 6: `ScriptProcessorNode` used to create Blocks like the `WhiteNoise` generator[24]

Some Blocks considered important to our environment and not present as Native Nodes were implemented using the Script Processor Node like ADSR envelop, AD envelop, White Noise Generator and explicit signal maths. None of these Blocks needed static parameters.

### HTML 5 and GUI

Our last set of Blocks included some HTML 5 widgets that compose the application’s GUI, like an object to print values on the screen, get mouse position and keyboard input, button, sliders, change background color and some other GUI features.

We also developed some Misc objects to convert MIDI to frequency, convert 3 float values into a RGB Color and convert char to float and float to char, for example.

All these objects were implemented using basic HTML and javascript code and without using other javascript library.

### Block Connections

Once we defined a set of Blocks, the next step on development was to create connections between Blocks. Audio input and output data could be easily connected since the Web Audio API uses this programming model to connect its Nodes.

The creation of other connections between objects, like char or float, was our first challenge. Our former implementation with OpenCV and C could use C pointers to pass values from one object to other and create Blocks connections. Since in Javascript there are no pointers, we had to look for another approach.

The solution used was to create an array of call-back functions to be called when some event occurs. This solution granted a responsive interface. Thus, when a GUI object action is performed, it can transmit the result to a list of connected Blocks and grant the value data flow.

## 5. Discussion

Our first set of audio Blocks were very experimental and had several issues to be solved. The

*Mosaiccode* had only C Blocks and we never tried to generate Javascript source code. Adapting the tool to another language meant to create Blocks, Ports, Connections and a Code Template to generate HTML + CSS + Javascript code. As explained before, to connect audio ports was simple but to create a interactive code with GUI elements was not simple.

The solution created solved the problem to our small set of Blocks and ports and it is extensible to every new developed Block. Since these issues were solved, to create several others Blocks is a more easy and trivial task.

Before starting the development of other Blocks, we performed a usability test with a group of users with audio development skills. This test could prove that a VPL/DSL really ease the development of Web Audio applications[25]. Also, it allowed naive users to create interesting sounds even without specific domain knowledge.

## 6. Conclusion

We present in this article a proposal to simplify the development of applications for the Digital Arts domain by means of the development of a set of Blocks to *Mosaiccode*, a visual programming environment that can be used to develop systems in specific domains. This set of Blocks were implemented using Javascript programming language and Web Audio API, allowing to develop cross-device application and enjoy powerful tools provided by web technologies.

Beyond the simplicity of the VPLs with DSL code reuse, developing the set of Blocks in the *Mosaiccode* enable the possibility of generating the source code of applications from VPL Diagram. This is the main difference between *Mosaiccode* and other VPLs for the Digital Arts domain like Pure Data or Max/MSP.

To define the set of Blocks, we investigated which Blocks could be necessary to the environment to satisfy the Digital Arts requirements, based on others tools like Gibberish and Pure Data. We started our Block development implementing the Web Audio Natives Nodes, then we implemented some Blocks considered important to our environment that are not present in Web

Audio Native Nodes using the the Script Processor Node. To complete the initial set of Blocks we included some HTML 5 widgets that compose the application's GUI and some Misc objects to convert values like MIDI to frequency.

This work also discusses the Block connection, presenting a way to connect them and expand the environment creating new components. We also cite an usability test performed before starting the development of other Blocks, that could confirm that a VPL/DSL really ease the development of Web Audio applications.

Combining web technologies with *Mosaiccode*, this work results in a set of Blocks that provides a quick, simple and practical way to develop cross-device applications to Digital Art domain.

Our Future works include expansion of the environment developing others APIs to the *Mosaiccode* like Web MIDI, WebGL, Canvas and SVG. We also intend to generate audio code for other programming languages.

## References

- [1] Charlie Roberts, Matthew Wright, JoAnn Kuchera-Morin, and Tobias Höllerer. Rapid creation and publication of digital musical instruments. In *NIME*, pages 239–242, 2014.
- [2] Charles Roberts, Graham Wakefield, and Matthew Wright. The web browser as synthesizer and interface. In *NIME*, pages 313–318. Citeseer, 2013.
- [3] Charles Roberts, Graham Wakefield, Matthew Wright, and JoAnn Kuchera-Morin. Designing musical instruments for the browser. *Computer Music Journal*, 39(1):27–40, 2015.
- [4] Ben Taylor and Jesse Allison. Braid: A web audio instrument builder with embedded code blocks. In *Proceedings of the 1st international Web Audio Conference*, 2015.
- [5] Stephane Letz, Sarah Denoux, Yann Orlarey, and Dominique Fober. Faust audio dsp language in the web. In *Proceedings of the Linux Audio Conference (LAC-15)*, Mainz, Germany, 2015.

- [6] Chris Wilson and Jussi Kalliokoski. Web midi api. <https://www.w3.org/TR/webmidi/>, 2015. Accessed: 201-09-30.
- [7] Paul E. Haeberli. Conman: A visual programming language for interactive graphics. *SIGGRAPH Comput. Graph.*, 22(4):103–111, June 1988.
- [8] Anabela Gomes, Joana Henriques, and António Mendes. Uma proposta para ajudar alunos com dificuldades na aprendizagem inicial de programação de computadores. *Educação, Formação & Tecnologias-ISSN 1646-933X*, 1(1):93–103, 2008.
- [9] Daniel D Hils. Visual languages and computing survey: Data flow visual programming languages. *Journal of Visual Languages & Computing*, 3(1):69–101, 1992.
- [10] R. C. Gronback. *Eclipse Modeling Project: A Domain-Specific Language (DSL) Toolkit*. Addison-Wesley, New York, 2009.
- [11] Marjan Mernik, Jan Heering, and Anthony M Sloane. When and how to develop domain-specific languages. *ACM computing surveys (CSUR)*, 37(4):316–344, 2005.
- [12] Arie Van Deursen and Paul Klint. Domain-specific language design requires feature descriptions. *CIT. Journal of computing and information technology*, 10(1):1–17, 2002.
- [13] Mark Danks. Real-time image and video processing in gem. In *ICMC*, 1997.
- [14] Miller Puckette et al. Pure data: another integrated computer music environment. *Proceedings of the second intercollege computer music concerts*, pages 37–41, 1996.
- [15] Matthew Wright, Richard Dudas, Sami Khoury, Raymond Wang, and David Zicarelli. Supporting the sound description interchange format in the max/msp environment. In *ICMC*, 1999.
- [16] Antonio Camurri, Shuji Hashimoto, Matteo Ricchetti, Andrea Ricci, Kenji Suzuki, Riccardo Trocca, and Gualtiero Volpe. Eyesweb: Toward gesture and affect recognition in interactive dance and music systems. *Computer Music Journal*, 24(1):57–69, 2000.
- [17] Colin BD Clark and Adam Tindale. Flocking: a framework for declarative music-making on the web. In *SMC Conference and Summer School*, pages 1550–1557, 2014.
- [18] Charles Roberts, Matthew Wright, JoAnn Kuchera-Morin, and Tobias Höllerer. Gibber: Abstractions for creative multimedia programming. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 67–76. ACM, 2014.
- [19] Victor Lazzarini, Edward Costello, Steven Yi, et al. Csound on the web. 2014.
- [20] John Resig, Ben Fry, and Casey Reas. Processing. js, 2008.
- [21] Yann Orlarey, Dominique Fober, and Stéphane Letz. Faust: an efficient functional approach to dsp programming. *New Computational Paradigms for Computer Music*, 290, 2009.
- [22] Anand Mahadevan, Jason Freeman, and Brian Magerko. An interactive, graphical coding environment for earsketch online using blockly and web audio api. 2016.
- [23] Mozilla Developer Network. Web audio api. [https://developer.mozilla.org/en-US/docs/Web/API/Web\\_Audio\\_API](https://developer.mozilla.org/en-US/docs/Web/API/Web_Audio_API), 2017. Accessed: 2017-06-30.
- [24] Mozilla Developer Network. Scriptprocessornode. <https://developer.mozilla.org/pt-BR/docs/Web/API/ScriptProcessorNode>, 2017. Accessed: 201-09-30.
- [25] Teste de usabilidade do sistema mosaicode. In *In: Simpósio Brasileiro de Sistemas de Informação, 2017, Lavras. Anais [do] IV Workshop de Iniciação Científica em Sistemas de Informação (WICSI)*, pages 5–8, Lavras - MG: Universidade Federal de Lavras - UFLA, 2017.

# Music Papers



# AirQ Sonification as a context for mutual contribution between Science and Music

Julián Jaramillo Arango<sup>1</sup>

<sup>1</sup> Sensor Laboratory – Design and Creation Program. Caldas University Calle 65 No. 26 – 10, Bloque Orlando Sierra, piso 4, 170004. Manizales, Colombia

julianjaus@yahoo.com

## Abstract

This paper addresses a high-level discussion about the links between Science and Music, focusing on my own sonification and auditory display practices around air quality. Grounded on recent insights on interdisciplinarity and sonification practices, the first sections will point out some potential contributions from scientific research to music studies and vice versa. I will propose the concept of mutualism to depict the interdependent status of Music and Science. The final sections will discuss air contamination as a complex contemporary problem, and will report three practice-based design projects, AirQ jacket, Esmog Data, and Breathe!, which outline different directions in facing local environmental awareness.

## 1. Introduction

Since the ancient world the concept of nature has linked Science and Music. However, in the 20th century modern physics and mathematics have promoted a shift of mind about the understanding of the surrounding reality and it has impacted music making. One example of this is composer Iannis Xenakis, whose notion of nature is not dependent on mimesis:

But it is important to note that the kind of nature he is referring to is the one supplied by modern science. It is far removed from, say, the naturalistic views proper to Classicism or Taoism – Nature as Harmony. This is fundamental to understand Xenakis' "naturalism". The nature he is referring to is the one of thermodynamics, of probabilities, of

Brownian movements, etc6. It was then normal that he expresses an interest in chaos theories, which were popularized only in the end of the 1970s and on. [1]

By grounding his music on modern physics and mathematical theories, Xenakis has contributed to the popularization of a shared basis about the answers that science provides to society.

This paper addresses a high-level discussion about the mutual relations between Science and Music, focusing on my own sonification and Auditory Display (AD) practices around air quality. I will discuss aspects of sonification and AD that can extend the scope of both, scientific research and musical studies. The paper is organized in four sections. In the first one, I will comment and compare arguments from two recent papers providing criteria to understand the logics of interdisciplinarity between Art and Science and the diverse directions that sonification projects can follow. In the second section, I will summarize the contributions from both ways. The third section is concerned with scientific and social aspects of air contamination, since it was an inspiring topic in the creation of three sonification projects: AirQ jacket, Esmog Data, and Breathe!. While reporting some of the project based research findings in the final section, I will discuss separately sonification and auditory display strategies adopted in the three projects.

The projects propose three different directions in facing local environmental awareness. They were created along with the Caldas University Design and Creation program in Manizales, Colombia, under a two-year postdoctoral research entitled Sound Design for

the Urban Space.

## 2. Mutualism between Music and Science

Sonification and Auditory Display practices are usually taken as examples of the interdisciplinarity between Music and Science. In her analysis of sonification projects dealing with natural phenomena, Alexandra Supper suggests “legitimacy exchange” to point out the rhetorical support that a scientific field can provide to another [2]. By analyzing the motivations of particular projects, Supper raises arguments about the positive roll of art in particular scientific endeavors. Moreover, she discusses the rationale of music creation and reception in artistic sonification.

Supper’s arguments take into consideration the study about the public understanding of art-science by Georgina Born and Andrew Barry [3], which reports an ethnographic research of art-science practitioners and institutions in the USA, UK and Australia. The authors claim a neglected diversity of the field; hence they propose accountability, innovation, and ontology as three different logics of the transdisciplinarity between Art and Science. The contextualization of the findings achieved inside the laboratory is an unavoidable stage in scientific endeavors dependent on public funding. By accountability Born & Barry “... refer to a series of ways in which scientific research is increasingly required to make itself accountable to society” [3]. Moreover, making scientific knowledge more accessible to the public contributes to the growth of modern knowledge economies. It relates to the logic of innovation, which draws “... attention to a range of arguments about the need for scientific research to fuel industrial or commercial innovation and economic growth” [3].

In the case of Sonification and AD, the logics of accountability and innovation suggest a biased partnership between Music and Science. As a recipient for the popularization of scientific knowledge, Music enriches research endeavors with alternative strategies to target the public sphere. Conversely, the contributions provided by Science to the field of Music will be found in the logic of ontology: “an

orientation in interdisciplinary practices towards effecting ontological change in both the object(s) of research, and the relations between research subjects and objects”. [3]

By extending the authors reflection, the biological metaphor of mutualism depicts the interdependent status of Music and Science in sonification practices. Although this concept is mainly concerned with living beings, mutualism refers to a particular relationship between Science and Music in which each field benefits from the activity of the other. In this regard, I will now summarize contributions from both, Science and Music. In other words, I will discuss in which aspects sonification and AD are extending the scope of both, scientific research and musical studies.

## 3. Summarizing mutual contributions

The topics I will discuss derive from two sources. The first one is the Supper’s analysis of scientific and artistic sonification projects. While it comes from a Science & Technology scholar, it seems a reasonable task counterbalancing some of these views from a musical composer perspective. The other source is my own sonification and AD research around air quality, which has advanced in a series of three creative projects. They will be described in the third section of this paper.

### 3.1 From Music to Science

Supper argues that the most valuable contribution of sonification is spreading scientific knowledge to the public. With this claim, she challenges the mainstream research in sonification and AD, which focus on sound as a scientific tool by taking advantage of sound’s non-visual, time-dependending and symbolic capabilities as a conveyor of data. While some sonification experiments are used on applications such as alarms, alerts, warnings, monitoring or status and process indicators, others relies on data exploration and “... use sound to offer a more holistic portrait of the data” [4]

Allied to the logic of innovation, scientific sonification projects provide alternative (to visual and verbal) approaches to



explore known data. Furthermore, under unpredictable, chaotic situations or those involving large amount of data, sonification is also explored in the search of patterns and new meanings. It is worth mentioning that in scientific sonification, musical content emerges as an involuntary product. Since low-level musical units such as notes, chords, melodies or even audio samples are regular material in the data-to-sound mapping processes [5], complex musical structures inevitably rise as a residue

Although Supper's assumption is not shared by the sonification and AD academic community, it does let see sonification contributions under the logic of scientific accountability. Artistic and musical experiments with sonification contribute to the public spreading and understanding of scientific knowledge. If a scientist finding that was achieved inside the laboratory should take effect outside his peers' context, knowledge accessibility should be taken into account from the research conception. Moreover, providing social meaning to scientific knowledge strengthens a shared basis about the answers and solutions that science provides to society.

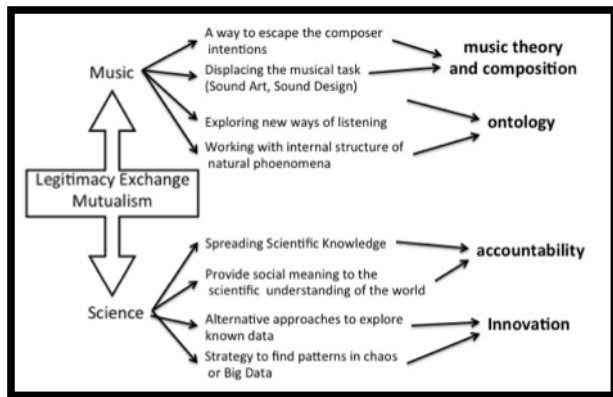
### 3.2 From Science to Music

The musical need of a data domain from which content could be generated arose when the second Viennese school disrupted the course of academic musical composition by deliberately rejecting the tonality and adopting twelve-tone techniques. While a generation of subsequent composers devoted to the integral or total serialism extended the ways in which the data domain was applied to music by mapping the series to diverse musical parameters, other composers explored modern mathematical models as data domains [6]. In this musical tradition, the case of composer Iannis Xenakis (1922-2001) remains as a remarkable example of shared aims between sonification and 20th-century musical composition. When discussing the Xenakis' interest for cellular automata, Makis Solomos raises concepts such as "fluid turbulence" and "automation" [1] as natural forces whose internal dynamics and emerging behavior gave rise to his orchestral work *Horos* (1986). Artistic works such as the *Storm*

*Sonification Project* (2004) [7] or the *Sonic Pavilion* (2009) [8] suggest a resumption of Xenakis' interest for the internal structure of nature as a data-domain. According to Supper, by allowing the spectator to witness phenomena that can otherwise only be perceived in the wild, at one's own peril, such as meteorological or telluric activity, artistic sonification can lead the listener to a overwhelming, sublime experience [2]. From an aesthetic perspective, the goal of sonification would be producing "...auditory representations that give insight into the data or realities they represent to enable inference and meaning making to take place". [9]. Sonification explores ways of listening not usually related to music, since it relies on the fourth Schaefferian mode or function, *Comprendre*, which comprises "...symbolic (i.e. consensual) relations between representamen and object" [10] and is activated when facing a "... structure that has a sense and meaning for those listeners who share the code" [11]. By triggering supplementary meanings to sound and creating auditory images of reality, sonification suggests an ontological transformation and novel interpretations of the encrypted data.

The displacement of the composer, listener and performer roll was one of the John Cage insights giving rise to sound art [12]. By adopting chance operations as the data domain, Cage was deliberately renouncing to his intentions as a composer, letting the world speak at its own pace and accepting its indeterminate content without creating expectations. This philosophical principle of 20th century music theory, composition and sound art is identified by Supper in *Sonification* projects as a "self-denying tendency on the part of many composers who are compelled by the possibility of removing the individual from the act of composition" [2]. While the data domain provides the content, letting the composer to put aside his intentions and expectations, the compositional decisions are then displaced to play an indirect roll in the process. In sonification and AD practices, the composer do have different creative tasks, most of them related to the representation of data, the mapping of the non-sounding information, the raising of auditory images of reality or the

construction of listening environments relevant to the sonified data. In many cases, artistic sonification endeavors are sound art installations, design projects, objects or systems rather than pieces of music.



**Figure 1: Summary of mutual contributions between science and music**

#### 4. The AirQ problem

With regard to climate change, the World Health Organization has warned about “the increasing temperatures, sea-level rises, changing patterns of precipitation, and more frequent and severe extreme events.” These phenomena “...are expected to have largely adverse effects on key determinants of human health, including clean air and water, sufficient food and adequate shelter” [12]. Monitoring air quality has become an important demand for any city council since pollution affects directly the population health. In our particular context an active volcanic region regularly emanates toxic gases surrounds Manizales. In addition, the city’s growing industry and vehicle fleet determine important factors for environmental contamination. Climate change irreversibility demands special attention to natural and human factors of contamination, suggesting that air pollution should become a critic element in the everyday life of local community.

In the “Clean Air Act”, the United States Environmental Protection Agency (EPA) established the Air Quality Index (AQI), considering a number of pollutant gases such as O<sub>3</sub> (ozone), PP (polystyrene particles), CO (carbon monoxide), SO<sub>2</sub> (sulfur dioxide) and NO<sub>2</sub> (nitrogen dioxide). For each of these pollutants, EPA has established air quality

standards to protect public health [13]. In Manizales Corpocaldas is the highest environmental authority being responsible for monitoring local air quality. They have placed air quality monitoring stations at different points in the city, focusing on SO<sub>2</sub> and NO<sub>2</sub> levels [14].

However, as Born and Barry state “... readings from these stations, since they are derived only from specific points, provide a very limited basis for estimating the state of the air mass over extremely large areas”. [3]. Although, there is significant available information about air pollution, environmental data are hardly taken into account by the community, since “...existing air quality monitoring regimes fail to address the question of how individuals interact with air, by breathing particular mixtures and quantities of pollutants during the course of their everyday activity in the city” [3]

From this point of view, air quality data became a rich source of information to be interpreted and its sonification an inspiring topic for sound creators, because it calls the attention of common citizens about environmental awareness. By dealing with environmental contamination as a problem to sonification and AD practices, “... air quality should not be considered a property of air, but understood as a relation between air and those who breathe and are affected by it...”[3]. It suggests an ontological transformation of both, the concept of air quality and the relation with local citizens.

#### 5. Project based research

Given the nature of the problem, I have adapted methodologies from Design studies, since this field proposes project-based directions for creative practices [15]. Design thinking considers both, the aesthetics and functional dimension involving modeling, interactive adjustment and re-design. The projects we will describe in this section, AirQ jacket [16], Esmog Data [17] and Breathe!, illustrate the “...distinction between the provision of public information and the practice of a public experiment” [3]. By testing different

AD and sonification proposals about the same environmental information, I have observed the implications and potentialities of each communication design. The iteration among the projects produced partial results and conclusions adopted as inputs in the sequence of prototypes.

I find an important difference between sonification, the data-dependent generation of sound and AD, which covers all the topics exploring the auditory channel to convey non-verbal information, including the display environment design (the audio system, speakers, listening facilities, etc...) [18]. While sonification invokes computer music research topics related to the creation of sonic content such as synthesis and DSP methods and representation of data structures, AD leads to other ones related to audio hardware design, diffusion and sonorization. In this regard, sonification and AD strategies are discussed separately.

The projects are not intended to be pieces of music, but objects and sound art installations that can be joined as sound-augmented consulting devices. They intend to bridge the gap between the empirical and scientific reading of reality, since they are in constant negotiation in the definition and understanding of our surroundings. Particularly, I am interested in calling the attention about the roll that air pollution plays in the everyday life of my local community, by enhancing environmental awareness through a rewarding listening experience.

### 5.1 AirQ Jacket (2016)

AirQ Jacket is a piece of clothing with an attached electronic circuit, which measures contamination levels and temperature and transforms this information to visual and acoustic stimuli. It was created along with fashion designer Maria Paulina Gutierrez whose participation in the laboratory triggered an interchange among electronic, sound and dressmaking crafting, which resulted in an unorthodox item. As a design's goal, the user should be able to acoustically consult our AD device wherever he/she goes, in order to create

healthy courses through the city.



**Figure 2: The AirQ jacket sonification system runs in a custom-made, stethoscopic sound artifact**

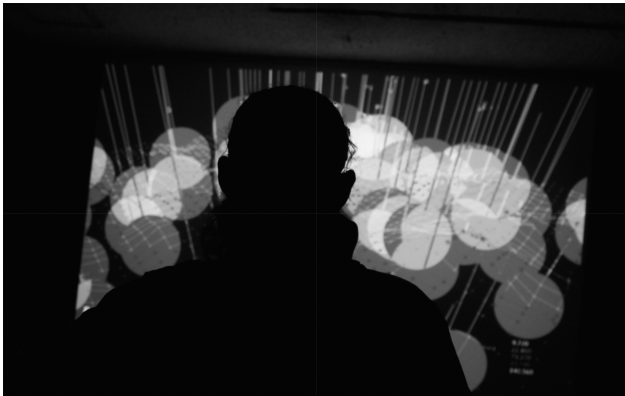
Since portability and lightness were challenges in the prototype of the AirQ jacket, the circuit was created involving the minimal specifications. We sewed a circuit with two affordable environmental sensors (Mq-135 and DHT-11) providing AirQ and temperature levels, an arduino and some leds. The AD system consisted of a custom-made stethoscopic artifact attached to the jacket and built with a piezo-electric device located inside a plastic cabinet that totally kills the sound, unless you approach it the ear.

We faced a challenge when creating sonic content with the arduino microcontroller, since it allows a meager repertoire of sound generation possibilities. We opted to work with a couple of sound pulses of energetic attack sounds. The first one displays temperature by varying speed and contamination level by varying pitch. The second pulse acts as a grid of reference, a contextual sound [4] representing “normal” environmental conditions.

### 5.2 Esmog Data (2016)

Esmog Data is an immersive installation presented in the Art Exhibition of the 2016 Balance-Unbalance Festival [19] along with Christian Quintero and Vanessa Gañán. The piece displays through audio and motion graphics the temperature and the concentration of some toxic gases determining air quality index (CO, CO<sub>2</sub>, SO<sub>2</sub> and PM<sub>10</sub>). The AD device comprises a surround speaker system

whose sonic content is constantly changing, since a custom-made environmental station located in the entrance of the exhibition space regularly refreshes the data.



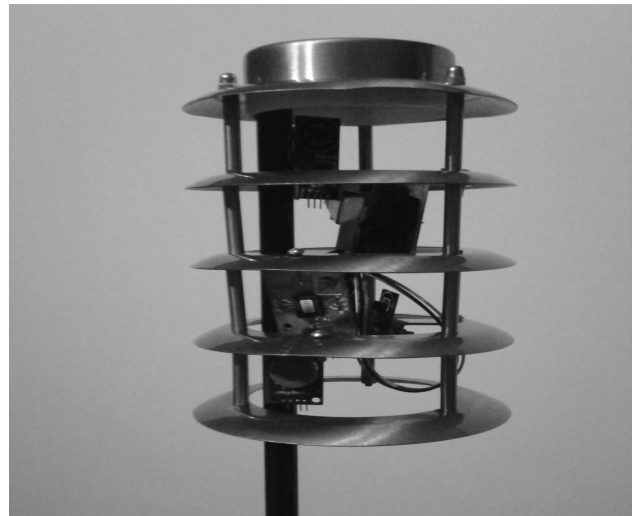
**Figure 3: Esmog Data runs in an immersive space.**

The sonic content was produced with the Johan Eriksson EcoSYSTEM [20] pure data patch, since it offered a modular workflow of audio synthesis blocks in which sensor data can be input. Rather than a monitoring system, the sonification design pursued triggering a meaningful listening experience. In this regard, instead of mapping sensor data directly to single synth parameters (pitch, amplitude, rate, adsr), we associated each one of the sensor inputs to many parameters in the synth blocks in the search of more complex musical values. Dennis Smalley vocabulary [21] was helpful to establish the qualities of sound motion attached to each pollutant gas. NO<sub>2</sub> was associated to the notion of unidirectional motion (*ascent, plane, descent*), PM<sub>10</sub> to occupation of the spectral space (*diffuseness-concentration*), CO to textural growth process (*agglomeration-dissipation*), CO<sub>2</sub> to multidirectional motion (*exogeny-endogeny*). Temperature was associated to the behavioral relationship of *dominance-subordination* among the toxic gases

### 5.3 Breathe! (2017)

BREATHE! is a multi-channel installation with no visuals, where the visitor should be able to identify each one of the measured toxic gases as a different sound source in the space. Since we are in the prototyping process we will discuss here some insights and motivations that

have been introduced as variables in the project-based research



**Figure 4: Breathe! air pollution station**

The installation displays six human breathing sound loops, which shrink and stretch according to toxic levels, from different points in the space. BREATHE! deliberately adopts a denotative sonification strategy, by dealing with human gestures as a attractors to call attention in the meaning making process. The human breathing acts both, as a collateral effect of pollution and as a source of pollution information. This insight was achieved by following vocal sketching rationale:

...despite differences in language and tonal range, the apparatus for producing sound is essentially similar for all people. This is beneficial for communication, as it ensures a good framework for understanding the intentions of the communicating partner. [22].

The installation comprises two modules. On the one hand, the AD device, which consists of a multi-speaker space through which the listener can walk in order to approach to each sound source. A more accurate CO<sub>2</sub> sensor, a NO<sub>2</sub> one and a Wi-Fi module were attached to an improved version of the air monitoring station prototype. Being located outdoors, in the exhibition building terrace, the environmental station update the data regularly. The two modules are connected in real time via network resources.

## 5.4 Discussion

The three above-described projects propose different solutions for a common challenge: making air quality data more familiar to local citizens, by producing a rewarding listening experience. The interweaved strategies on auditory display and sonification that we will describe seek to face the project's goal.

### 5.4.1 Auditory Display

AD design is concerned with the situation in which the listener will make the consultation and the context in which he will experience the data. In this regard, the AirQ jacket stethoscopic artifact, that can be consulted while the user is in a city course, is intended to generate a pragmatic informational link between the user and data. As well as other 19th-century auditory devices the stethoscope produces objectification, which is the "... capacity to make external and concrete, and hence situate as perceptually objective" [23]. For its part, Esmog Data is intended to produce a sense of immersion by deploying the surround sound and the motion graphics systems in a dark room. As Supper remarks, this sense of immersion rarely depends on a single sense, but rather on the interplay of the senses [2]. Lastly, Breathe! Intends to elicit an interactive relation with sound. The listening space, a room with dimmed light rounded by six speakers where the user can walk and approach to the sound sources, invoke to an exploratory experience through which the subtle differences among breathe audio samples can be figured out.

	AirQ Jacket	Esmog Data	Breath
<b>AD</b>	Stethoscopic device	Surround 2.1	6-channel speaker system
<b>Experience</b>	Objective	Inmersive	interactive
<b>Sonification</b>	Reference sound	Abstract musical Values	Vocal sketching
<b>Listening</b>	comparing	appraising	proprioception

**Table 1: Sonification and auditory display strategies summary.**

### 5.4.2 Sonification

Sonification design is concerned with the mapping, the creation of the sonic content and the listening strategies needed in deciphering the information encrypted in sound. The AirQ

jacket sonification design demands from the user a comparison between two pulses of attack sounds, from which he has to be able to discriminate simple musical values, such as sound pitch and pulse speed. As other technical listening attitudes, it is expected that the user only will feel comfortable with the system after a learning period. As it was mentioned, the Esmog Data sonification design seeks to elicit an interest for sound and produce a meaningful listening experience. In this regard, the installation visitor will have to gather the environmental data from complex musical values, such as sound motion and texture qualities, what demands a deep sound appraisal effort. Finally, by sketching sounds with the voice, in Breathe! it was possible to observe sound's emotional and communicative aspects. Breathe! sonification design invokes the multimodal aspects of sound meaning by dealing with the tension and relaxation of muscles and activating a sort proprioceptive listening [11]. As a result, the sonic content evokes an acoustic reference of the natural pulses of the human body being affected by contamination.

### Acknowledgments

This work is the result of a research funded by Departamento Administrativo de Ciencia, Tecnología e Innovación (COLCIENCIAS), Grant 2014-656.

### References

- [1] Solomos, M. (2005) "Cellular Automata in Xenakis' Music Theory and Practice". In: Solomos, M., Georgaki, A., Zervos, G. (eds.), *Definitive Proceedings of the International Symposium Iannis Xenakis*. Athens
- [2] Supper, A. (2014) "Sublime frequencies: The construction of sublime listening experiences in the sonification of scientific data." *Social Studies of Science*. Vol 44(1) 34-58
- [3] Born G and Barry A (2010) *Art-science: From public understanding to public experiment*. *Journal of Cultural Economy* 3(1): 103-119.

- [4] Walker, B. & Ness, M.A. (2011). Theory of Sonification. In Herman, T., Hunt, a., Neuhoff, H. (ed) The sonification Handbook. Logos-Verlag, Berlin.
- [5] Grond, F. & Berger, J. (2011) Parameter Mapping Sonification. In Herman, T., Hunt, a., Neuhoff, H. (ed) The sonification Handbook. Logos-Verlag, Berlin.
- [6] Mc Laughlin, S. (2016) Composers and Chaos: A Survey of Applications of Chaos Theory in Musical Arts and Research. In Handbook of Applications of Chaos Theory p. 893–912;
- [7] Polli A (2005) Atmospherics/weather works: A spatialized meteorological data sonification project. *Leonardo* 38(1): 31–36
- [8] Atkins, D. (2009) Sonic Pavilion. Retrived from. <http://www.inhotim.org.br/inhotim/arte-contemporanea/obras/sonic-pavilion/>
- [9] Barras, S.&Vickers, J. (2011). Sonification Design and Aesthetics. In. Herman, T., Hunt, A., Neuhoff,H. (Ed). The sonification Handbook. Logos-Verlag, Berlin.
- [10] Palombini, C. (1999). *Musique Concrète Revisited*. *Electronic Musicological Review*, 4. UFPr Arts Department
- [11] Smalley, D. (1996) “The listening imagination: Listening in the electroacoustic era”, *Contemporary Music Review* 13 (2), 77-107
- [12] Charles, D & Cage, J. (2000) *For the Birds*. Paperback. NY
- [13] Ministry of Sustainable Development. (2017). Air Quality Index. Environment and the fight against the climate change. Retrieved <https://www.airnow.gov/>
- [14] CORPOCALDAS. (2016) Regional Environmental Plan Management REPM 2007-2019. Retrieved from <http://www.corpocaldas.gov.co/>
- [15] Findeli. A (2008) *Research Through Design and Transdisciplinarity: A Tentative Contribution to the Methodology of Design Research*. I: Focused -- Current Design Research Projects and Methods, Publisher: Swiss Design Network, Editors: Swiss Design Network, pp.67-91
- [16] AirQ Jacket (2016). <https://sonologiacolombia.wordpress.com/lab/airq-jacket/>
- [17] Esmog Data (2016). <https://sonologiacolombia.wordpress.com/lab/esmog-data/>
- [18] Hunt, A. And Hermann, T. (2011) *Interactive Sonification*,. In Herman, T., Hunt, A., Neuhoff, H. (ed) The sonification Handbook. Logos-Verlag, Berlin.
- [19] Balance-Unbalance 2016. <http://www.balance-unbalance2016.org/>
- [20] Eriksson, J. *EcoSYSTEM*. <http://www.monologx.com/ecosystem/>
- [21] Smalley, D. (1997) “Spectromorphology: explaining sound-shapes”, *Organised Sound*, 2(2),107-26.
- [22] Ekman, I., & Rinott, M. (2010). Using vocal sketching for designing sonic interactions. In *Proceedings of the 8th ACM conference on designing interactive systems* (pp. 123-131). ACM.
- [23] Rice T (2008) “Beautiful murmurs: Stethoscopic listening and acoustic objectification”. *The Senses & Society* 3(3): 293–306

# An analysis of *Desdobramentos do continuo* for violoncello and live electronics performed by audio descriptors

Danilo Rossetti<sup>1</sup>, William Teixeira<sup>2</sup>, Jônatas Manzolli<sup>1</sup>

<sup>1</sup> First Laboratory – Interdisciplinary Nucleus for Sound Communication of UNICAMP  
Rua da Reitoria, 165 - “Cidade Universitária Zeferino Vaz” – 13083-872, Campinas, SP

<sup>2</sup> Second Laboratory – Federal University of Mato Grosso do Sul  
Avenida Costa e Silva, 79070-900, Campo Grande, MS

danilo.rossetti@nics.unicamp.br, william.teixeira@ufms.br, jotamanzo@gmail.com

## Abstract

This article proposes an audio descriptors model for the analysis of live electroacoustic music. In this context, an analysis of the work *Desdobramentos do continuo* for violoncello and live electronics is addressed, concerning both tape (deferred time) sounds and live electronics (instrument sound and real-time processing). For this analysis, audio descriptors such as spectral flux, energy mean, centroid, and loudness were employed. The objective was to define which events produce huge timbre variations and to identify timbre subtle nuances which are not perceptible on a first listen of the work. We conclude comparing the analysis results to the compositional hypotheses presented in sections 2 and 3.

## 1. Introduction

Audio parametric descriptors are tools which extract different information from audio recordings. The objective of this procedure is to analyze these data aiming to understand some features related to human auditory perception and to perform a classification of the evaluated pieces and musical styles. This research area is known as MIR (Music Information Retrieval) and the obtained analyses results until now are available in the MIREX web page<sup>1</sup> (Music Information Retrieval Evaluation eXchange).

The use of audio descriptors for musical classification was already employed in previous researches, such as in Pereira [1], Peeters [2] and Peeters *et Al.* [3]. The Interdisciplinary Nucleus for Sound Communication of

UNICAMP (NICS) developed similar research in the past few years, resulting in the works of Monteiro [4], and Simurra; Manzolli [5,6]. In relation to the use of audio descriptors specifically for the analysis of contemporary music, we mention the work of Malt and Jourdan [7].

The general objective of this article is to contribute in the area of analysis of live electroacoustic music by audio descriptors. Specifically, an analysis of Rossetti's work *Desdobramentos do continuo* (2016), for violoncello and live electronics is performed. We first present the work contextualization and a discussion about the instrumental extended techniques employed in the instrumental writing. For the analysis, parametric audio descriptors are applied to the audio recording. These audio descriptors form an analysis model which can possibly be used to analyze other live electronics works. Our analysis will be centered on the behavior of the spectral flux, energy mean, centroid and loudness descriptors whose definitions will be detailed further up.

The reason for the use of audio descriptors in this analysis is that the employed instrumental extended techniques produce new sounds and transients. In this sense, an analysis based only on the score do not encapsulate the entire possibilities of the work. Furthermore, the interaction between the fixed tape sounds and the dynamic part of the sound processing produce dynamic elements which are beyond the score notation. Thus, to contemplate these aspects related to the work, a new methodology of analysis is necessary and will be developed next.

<sup>1</sup> [http://www.music-ir.org/mirex/wiki/MIREX\\_HOME](http://www.music-ir.org/mirex/wiki/MIREX_HOME)

## 2. Work Contextualization

*Desdobramentos do contínuo* is a work for violoncello and live electronics composed in 2016 by Danilo Rossetti. It is the last work included in his doctoral thesis [8], which investigates interaction and convergence possibilities between acoustical instruments and electroacoustic treatments [9]. This work is dedicated to the cellist William Teixeira, who participated in its development, which involved rehearsals, cello recordings, and audio analyses.

The general form of the work contents two parts that differ from each other mainly concerning the employed electroacoustic treatments. These treatments can be implemented in real time (morphological transformations of the cello sound captured live along the performance) or in deferred time [10] (audio manipulations involving phase vocoder and convolution processes from pre-recorded cello phrases).

Among the real-time treatments, granulation, microtemporal decorrelation, and dephasing are employed. An ambisonic spatialization of the electroacoustic sounds is conceived, creating a diffused sound field that surrounds the listener in performance. This spatialization is planned for an octophonic model, however quadrasonic and stereo versions of the piece also can be performed. The integration of real-time electroacoustic treatments with an ambisonics spatialization is achieved by the utilization of the *process~object*, belonging to the High Order Ambisonics Library (HOA). This library was developed by the CICM of *Université Paris 8* [11]. In *Desdobramentos do contínuo*, the architecture of the patch was implemented in Max MSP.

The objective of overlapping both deferred time sounds and real-time treatments of violoncello sound was to explore different possibilities of the electroacoustic universe. The adopted compositional hypothesis was that this combination would be complementary in terms of sound morphology. So, the overlapped sounds and would merge together into a single timbre. In this process, the tape sounds have a continuous and similar development, on the other hand, real time treatments generate sounds with discontinuous granular characteristics. In the analysis performed further on, these

questions will be verified.

Next, the instrumental part of *Desdobramentos do contínuo* will be discussed, focusing on extended techniques and the resultant sound morphology.

## 3. Instrumental extended techniques

The role of instrumental writing within the discourse<sup>2</sup> [12] of the piece is immense, but perhaps not in the way expected from a piece regularly written for an acoustic instrument with electronic support. The concept of the piece started in an attempt to escape from two extremes usually noticed in pieces written within the genre.

On the one hand, a compositional tendency is identified where the instrument functions simply as a signal generator, with the electronic synthesis being the most prominent character of the development of musical discourse. The instrumental gesture functions almost in subordination to the electronic gesture and the later function is to constitute continuity to the insertions, almost disparate, of the former.

On the other hand, it is possible to notice another extreme, where the instrumental gesture alone assumes the role of the generator of musical material, and the electronic support working just like that, a support, a kind of effects box that only ornaments the musical discourse, almost autonomous, executed by the acoustic instrument. In this case, electronics create only small inserts of effects, and sometimes act like a tape, executing another set of materials without any interaction with the instrumental gesture.

*Desdobramentos* comes from this attempt to overcome such extremes, starting from the interaction between the two sound sources as basic writing material. It is important to state that, because, in this sense, the instrumental writing works in a way not so much “soloistic”, but much more like chamber music, since musical materials are generated by both and often, from the interaction of both. At all times there is a feeding of new gestures, where it is up

---

<sup>2</sup> Musical discourse is a term adopted here to mean the whole relationship between musical agents and not as a synonym of musical work and even less to notation.



to the instrumentalist to be able to respond instantaneously to the stimuli produced by the electronic source, including the sort of sound produced by extended techniques; like chamber music, these stimuli never repeat themselves, because they are also responses, in turn, of events previously produced by the instrument and which are never identical. This is the great beauty and difficulty presented by the proposal and that ends up giving a great dynamism to musical discourse. Understood in this totality, the discourse assumes more fully its vocation of interacting with, for and through its agents.

Even so, the instrumental writing brings difficulties of a very advanced technical level that needs to be overcome for the effectiveness of the mentioned interactions. One of the first questions that rise when the musical score is performed is the presence of three levels of bow pressure, as in the following passage, Figure 1:

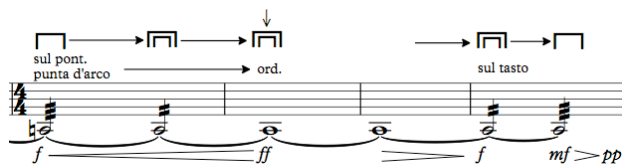


Figure 1: Measures 31 to 34 of the score

Although the performance of this kind of sonority is already very settled within the writing for strings in the contemporary repertoire, the piece brings a new issue that is the execution of long passages in *legato* with these different levels. This requires more than just changing the 90° angle of the bow in relation to the string, which usually generates a distorted sound, but it requires the extra weight of the interpreter. Making the three levels sound distinctive and at the same time homogeneous throughout the piece results in the fact that the piece requires a lot of the musician's physique when played in its entirety.

Another very demanding gesture it is in the “rebounds section”, so to speak, where different kinds of bow-strokes are prescribed in different rhythmical structures and with different numbers of notes per stroke, as in Figure 2:

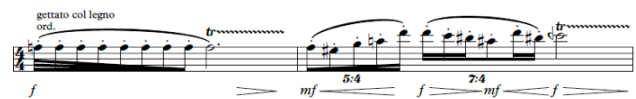


Figure 2: Measures 43 and 44 of the score

The aim here is to make the sounds always respond to the granular sound in the electronic synthesis, so the duration of each note must be at the same time proportional to the duration written, fluent in the gestural flow and as short as a granular sound must be to put the sound inside the bigger sonority.

The last passage that worths to be mentioned for its odd instrumental technique is this in Figure 3, but that occurs in other sections in the end of the piece:



Figure 3: Measures 43 and 44 of the score

This is a good example where traditional technique must be expanded not because a different timbre is required, but due to a new musical context; here a regular rush passage is full of notes written in hard string skipping and in the same direction of the bow, everything in a *crescendo* gesture. The result of such requirements among the microtonal pitches is an only and single sonority, almost like the Mannheim rockets in the Classical period, but that reviews the prominence of sound instead of only notes.

#### 4. Audio Descriptors Model

To analyze *Desdobramentos do contínuo*, a model formed by different types of audio descriptors was determined. This model included descriptors which provide temporal, spectral, energy, and psychoacoustic features. The selected descriptors (detailed below) were spectral flux [1,4], energy mean (RMS) [1,4], spectral centroid [1,2,3,4], and loudness [1,2,3,4]. The computational environment used for the descriptors calculation was the *Pdescriptors Library*, designed by Adriano Monteiro in PureData software [4], and revised by Gabriel Rimoldi.

Spectral flux descriptor is defined as the

magnitude difference between two successive analysis windows (frames). This descriptor provides lower values when the spectrum remains relatively invariable, on the other hand, it provides higher values when huge variations between successive frames are found. It is calculated from the expression below:

$$f\hat{e}_i = \sum_{k=1}^K \{\log_{10}[X_i(k)] - \log_{10}[X_{i-1}(k)]\}^2$$

Energy mean or RMS (root mean square) is the root mean square of amplitude values in a window analysis. These values describe the energy envelope profile of a sound and are defined by the following equation:

$$RMS_i = \sqrt{\frac{\sum_{n=0}^{N-1} x_i[n]^2}{N}}$$

The spectral centroid is the barycenter of the energy distribution belonging to the spectral envelope of a sound. Perceptively it is related to the sound brightness perception. Higher values characterize the predominance of high frequencies in the signal (in Hertz) and lower values characterize the predominance of lower frequencies, in terms of energy. The spectral centroid is calculated from the following expression:

$$ce(i) = \frac{\sum_{k=1}^k k * |X_i(k)|^2}{\sum_{k=1}^k |X_i(k)|^2}$$

Loudness is a psychoacoustic measure related to the perception of sound amplitude. It is variable according to different frequency bands (as demonstrated by the Fletcher and Munson curves, in 1933) and describes the auditory sensation of amplitude variation of a given sound. The loudness of a spectral analysis window is determined by this equation, according to Pereira's model [1]:

$$L_i = \sum_{k=1}^{k-1} |X_i[k]|^2 10^{(W[k])/20}$$

These chosen audio descriptors will be applied to the audio recording of the piece whose analysis will be presented next.

## 5. Analysis by Audio Descriptors

In the audio of the performance used for

this analysis<sup>3</sup>, the entire piece lasts 10'35''. Its first part goes from the beginning to 4'47'', and the second part from 4'47'' to its ending. In the first part, the deferred time process corresponds to a phase vocoder that stretches the spectrum of a given sound and repeats it continuously as a loop. During this part, the sound is sent to a granulator which has six different presets containing a sort of parameter values (such as grain size and rarefaction). These presets determine the direction of the sound mass evolution, whose perception changes gradually from a continuous timbre to a grainy sound cloud considerably rarefied.

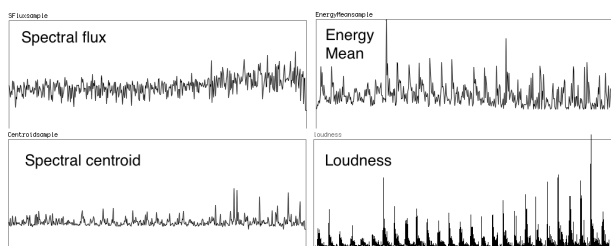
In the second part, the applied deferred time process to generate the tape sounds was the convolution between different pre-recorded cello sounds. In total five sounds of different durations were generated by this process (which have respectively 35'', 26'', 50'', 62'' and 78'' of duration). As common perceptual features among them, all these sounds have continuous spectral evolutions in time. It is important to remark that during the entire piece, besides the tape sounds, the cello sound is granulated in real time (its parameters are constantly modified), and the electroacoustic timbre (formed by these layers) is spatialized through high-order ambisonics models.

### 5.1 Analysis of Deferred Time Sounds

In this section, the looped phase vocoder sound of the first part (which changes gradually in time) and the five tape sounds of the second part, generated by convolution processes, will be analyzed and discussed.

In the first part, the phase vocoder generated sound evolves directionally from a continuous texture to a grainy sound mass which gradually becomes more discontinuous in perception. Our audio descriptors model was applied to the audio and, the resultant graphics are presented in Figure 4.

<sup>3</sup> The performance took place in NICS UNICAMP at 16<sup>th</sup> December 2016, by Pedro Bortolin (cello) and Danilo Rossetti (live electronics).



**Figure 4: Descriptor model applied to phase vocoder sound**

As it is showed in the graphics above, energy mean evolution (RMS) presents few peaks of energy which appear periodically. Otherwise, the centroid curve has also fewer and weaker peaks in the beginning. It just has relatively stronger peaks in the end of sound evolution. From these observations, we assume that, in average, both of these curves have no considerable variations.

On the other hand, despite the strong peaks belonging to the loudness curve and the smaller peaks belonging to the spectral flux curve, both of them raise in intensity at their end. At the same point (around 3'10"), spectral flux and loudness curves both start to have higher values. Here, the growth of the spectral flux curve is more consistent, since it is more constant, meaning that there are more intensity variations between successive frames. In loudness curve, the perceived increase (from 3'10") is mainly related to the periodically existent peaks. After them, the intensity sensation falls to lower levels.

We assume that the variations found in spectral flux and loudness curves are related to the granulation parameters applied to the phase vocoder sound. In the first part of the work, six different granulation preset values were applied. On them, while the feedback rate and grain delay remain constant, the grain size decreases from 400 to 75ms, while rarefaction rate increases from 0 (a totally continuous sound in perception) to 0,8 (indicating 80% of silence in the diffused sound mass).

In relation to the grain cloud perception, it is important to emphasize that bigger grains generate sonorities that privilege the sustained parts of the sounds (normally characterized by the presence of a fundamental frequency and upper partials). Smaller grains have a prominent presence of attack transients. For this reason, from a sound morphology standpoint, grain

clouds formed by smaller grains sizes (of less than 100ms) have a noisier auditory perception [8,9].

On the second part of *Desdobramentos* five tape sounds were addressed (Seq. 1 to 5). As a common feature of all these tape sounds they all have a continuous evolution. However, it is desired to investigate if they have different evolution characteristics. In this sense, audio descriptors can help us to evaluate some timbre features of these sounds, in order to describe their behavior. We applied the presented descriptors model to each sound and extracted the normalized (from 0 to 1) arithmetic average of each descriptor value (Table 1). This strategy was adopted to obtain significant data, in order to compare the evolution of the descriptors applied to the sounds.

Sound/ Descriptor	Seq. 1	Seq. 2	Seq. 3	Seq. 4	Seq. 5
Flux	0,32	0,4074	0,3111	0,2733	0,3051
RMS	0,183	0,3612	0,2962	0,4797	0,4094
Centroid	0,084	0,3515	0,2388	0,2581	0,3816
Loudness	0,63	0,6051	0,7636	0,7291	0,7113

**Table 1: Normalized averages of the audio descriptors of the second section**

From Table 1, it is possible to verify that the five sequences of the deferred time sounds show a gradual increase of energy means. This behavior is more prominent in Seq. 4 and 5. Therefore there is more spectral energy at the end of the piece. These spectral changes act in the perception as an increase of intensity and sound density.

Regarding spectral centroid values, we observe that the five tape sounds are organized in three brightness levels: low, middle and high. The low brightness level is assigned to the Seq. 1, the middle brightness level is assigned to Seq. 3 and Seq. 4, and the high brightness level is related to Seq. 2 and 5.

Finally, the normalized loudness average values are centered in a middle-high level. Seq. 1 are nearer to the middle level, while Seq. 3, Seq. 4 and Seq. 5 have higher intensity perception. Next, in Figure 5, a histogram graphic is presented, showing the descriptors average values related to each tape sound, in complementation to Table 1.

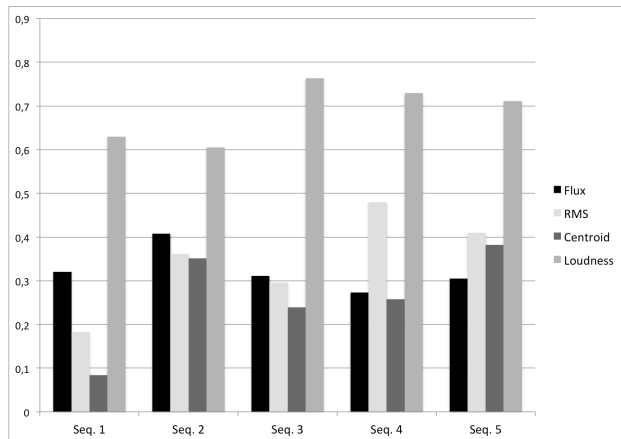


Figure 5: Descriptor values for each sound

Taking into account this figure, some observations can be made, in consideration to a global view of the descriptor values of each sound. Seq. 1 has the lower energy, lower centroid and one of the lowest loudness values. Seq. 2 has the higher flux, high centroid, and the lower loudness. Seq. 3 has the higher loudness, average centroid, and average-low energy. Seq. 4 has the higher energy, high loudness, average centroid and the lower flux. Seq. 5 has the higher centroid value, high energy, and low flux.

### 5.1 Analysis of Real-Time Processing

In this analysis, we focus on the real-time granulation of the violoncello sounds merged together with its acoustical sound. For this purpose, two different excerpts of the piece will be addressed. These excerpts were chosen from a spectral flux analysis of the entire piece. They correspond to moments where the flux descriptor reaches high values (over the average).

As we can see in Figure 6<sup>4</sup> (sonogram and spectral flux curve of the entire piece), there are more than two moments in the graphic presenting high flux values. Then, to decide which excerpts would be addressed, we opted to take those in which we have distinct instrumental techniques performed. It is known that different instrumental techniques generate different timbre morphologies. So, it becomes possible to compare the average descriptor

<sup>4</sup> In order to have both the sonogram and the spectral flux curve in the same figure, this audio analysis was performed in Sonic Visualiser software.

values correspondent to these morphologies (instrument and real-time electronic sounds).

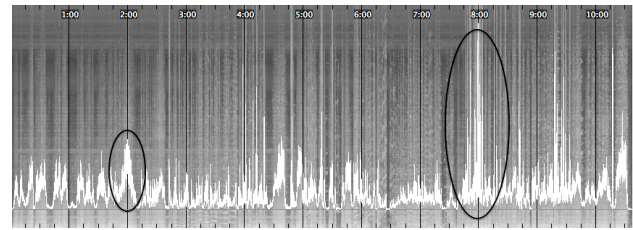


Figure 6: *Desdobramentos do contínuo* sonogram and spectral flux curve: two circulated excerpts to be analyzed

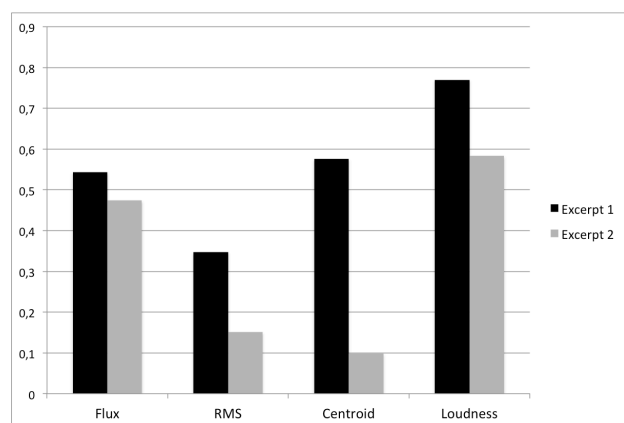
The first excerpt corresponds to measures 31 to 34 of the score, shown in Figure 1 (1'55'' to 2'10'' of the audio recording). As it is observed in this figure, there is a C2 pitch which is sustained by four measures, having as features different speeds of *tremolo*, different bow positions (*ordinario*, *sul ponticello* and *sul tasto*), and different bow pressures over the string (normal pressure, exaggerated and creaking noise). As the combination result of these instrumental possibilities, different timbres were generated.

In addition, the descriptors model (spectral flux, energy mean – RMS –, centroid and loudness) is applied to the correspondent audio of this part. The arithmetic average of the achieved normalized values was also calculated, in order to make possible the comparison with the average values of the second example.

The second excerpt refers to measures 119 to 123 of the score (7'42'' to 8'05'' of the audio recording), shown in Figure 7. In this example, the main instrumental techniques played by the violoncello are closer to percussive gestures (*pizzicatti*, *pizzicatti Bartók*, and *gettato col legno*), generating discontinuous sonorities. *Pizzicatti tremolo* and bow positions from *ordinario* to *sul tasto* are also requested in the score in this excerpt.

Figure 7: Measures 119 to 123 of the score

In the score (Figure 7), the number “12” above measure 119 indicates the Max patch preset which corresponds to specific values of the cello granulation. It also indicates the beginning of the fourth tape sound belonging the second part of the piece, which was previously analyzed. The same audio descriptors model was applied to this excerpt and their arithmetical averages are shown below in Figure 8 graphic. In this graphic, a comparison between values of excerpts 1 and 2 is performed.



**Figure 8: Descriptor average values of excerpts 1 and 2**

Regarding the Figure 8 graphic, the average values of spectral flux descriptor for both excerpts are relatively close. Loudness level is a little higher for the first excerpt, in comparison to the second.

Higher differences were found in RMS and centroid descriptors, which may be attributed to the morphological characteristics of the generated timbre. The first excerpt contains continuous timbre variations of a C2 pitch, according to the bow pressure and position, and also to the presence of the *tremolo* effect. We observed that the intensity has a constant behavior in this example, due to a variation from *forte* to *fortissimo* in the score notation. In relation to the real-time granulation response, it is correlated in most aspects to the acoustical instrument sound morphology.

Average RMS values of both excerpts are very different. It can be attributed to the timbre prominent discontinuity of the second excerpt, characterized by instrumental techniques which produce “percussive” sonorities. Moreover, the average values corresponding to centroid have a

huge difference. It means that the spectrum barycenter (brightness sensation) of the first excerpt is located in an average-high area, on the other hand in the second excerpt it is located in a low-frequency area.

Thus, from these two excerpts, we observed that different instrumental techniques combined with real-time granulation (having both high spectral flux values) differ each other mainly from centroid and RMS average values. Otherwise, the average loudness values were not considerably different, reflecting on similar auditory sensation levels.

## 6. Conclusion

In this article, we intended to establish an audio descriptors model which is useful for the analysis of live electroacoustic music. This is an attempt to contribute in the field of computer-aided music analysis. For the analysis of *Desdobramentos do contínuo*, we found relevant the use of spectral flux, energy mean, centroid and loudness descriptors. For further analyses or even to a more detailed analysis of this work, it is possible to increase our model with new descriptors.

From the analysis of tape (deferred time) sounds by these descriptors, we could extract important information that clarifies some characteristics of the timbre of these sounds. On a first listen, we tend to consider them similar to each other, due to their continuous time evolution. However, after the application of our descriptors model, subtle variations become noticeable and our perception becomes more attentive to these nuances. It is also desirable during the performance that the interpreters are aware of these nuances. Thus, they can interact with them on a higher level, in order to produce a more convergent performance, considering acoustic and electroacoustic parts.

These subtle timbre variations, in a certain way, complement the previously presented compositional hypothesis. The tape sounds have a global continuous evolution. However, for the phase vocoder sound, after a certain point, there is a discontinuity perception demonstrated by higher flux and loudness values. In relation to the five tape sounds of the second part, variable values of RMS and

centroid indicate different features of their global timbre perception.

In addition, despite the tape sounds nuances, the main timbre differences in the global perception of the work (defined by the variations of the spectral flux descriptor applied to the entire piece) are related to the employed instrumental techniques and their real-time granulation. Considering other audio descriptors, these timbre variations reflect mostly on RMS and centroid differences.

Thus, from this analysis, we verified that the change in the perceived timbre morphology is mostly guided by the instrumental writing. However, in real-time electroacoustic music (in general) this causality can be broken by the employment of tape sounds with unexpected effects and events.

## 7. Acknowledgments

Danilo Rossetti is supported by FAPESP in his post-doctoral research at NICS-UNICAMP (process 2016/23433-8). Jônatas Manzolli is supported by CNPq under a Pq fellowship, process 305065/2014-9.

## References

- [1] Erica M. Pereira. Estudos sobre uma ferramenta de classificação musical. Dissertação (Mestrado). FEEC Universidade Estadual de Campinas, Campinas, 2009.
- [2] Geoffroy Peeters. A Large Set of Audio Features for Sound Description (Similarity and Classification) in the CUIDADO Project. Cuidado Project Report. Institut de Recherche et de Coordination Acoustique-Musique (IRCAM), 2004.
- [3] Geoffroy Peeters *et Al.* The Timbre Toolbox: Extracting audio descriptors from musical signals. In *J. Acoust. Soc. Am.* 130 (5), November, 2011, pages 2902-2916.
- [4] Adriano C. Monteiro. Criação e performance musical no contexto dos instrumentos musicais digitais. Dissertação (Mestrado). Instituto de Artes, Universidade Estadual de Campinas, Campinas, 2012.
- [5] Ivan Simurra; Jônatas Manzolli. Sound Shizuku Composition: a Computer-Aided Composition Systems for Extended Music Techniques. In *MusMat Brazilian Journal of Music and Mathematics* 1 (1), Rio de Janeiro, 2016, pages 86-101.
- [6] Ivan Simurra; Jônatas Manzolli. O azeite, a lua e o rio: o segundo diário de bordo de uma composição a partir de descritores de áudio. In *Revista Música Hodie* 16 (1), Goiânia, 2016, pages 101-123
- [7] Mikhail Malt; Emmanuel Jourdan. Zsa.Descriptors: A Library for Real-Time Descriptors Analysis. In *SMC 08 Sound and Music Computing Conference*, Berlin, 2008.
- [8] Danilo Rossetti. Processos microtemporais de criação sonora, percepção e modulação da forma: uma abordagem analítica e composicional. Tese de Doutorado. Instituto de Artes, Universidade Estadual de Campinas, Campinas, 2016.
- [9] Danilo Rossetti. Interaction, Convergence and Instrumental Synthesis in Live Electronic Music. In: M. Aramaki, R. Kronland-Martinet, S. Ystad, editors, *Proceedings of the 12<sup>th</sup> CMMR*, pages 209-216, The Laboratory of Mechanics and Acoustics, 2016, pages 209-216.
- [10] Flo Menezes. Fusão e contraste entre a escritura instrumental e as estruturas eletroacústicas. In Flo Menezes, *Atualidade estética da música eletroacústica*, Editora UNESP, pages 13-20, 1999.
- [11] Pierre Gillot. Documentation de la bibliothèque HOA: Les champs sonores. CICM, Université Paris 8, Labex Arts H2H. <http://www.mshparisnord.fr/hoalibrary/ambisonie/les-champs-sonores/>
- [12] William Teixeira. O discurso musical. In: F. Presgrave; L. Noda; J. J. Mendes, editors, *Ensaio sobre a música do século XX e XXI: composição, performance e projetos colaborativos*, EDUFRRN, pages 226-243, 2016.



# Communicating a World View: *figer*, a Manifold Composition

Sever Tipei

Computer Music Project, University of Illinois  
Urbana, Illinois. USA

s-tipei@illinois.edu

## Abstract

A composition, *figer*, for computer-generated sounds, is analyzed in some detail. The formal architecture, and types of materials used are discussed along with particular features of DISSCO, the original software employed. A non-linear narrative is detected and the implications of an work open to multiple interpretations are examined together with existing clues about the author's belief system. Finally, the world view embedded in the composition is analyzed and the merits of comprehensive or "black box" software are identified as essential to the production of composition classes or *manifold compositions* such as *figer*.

## 1. Introduction

In the words of Arthur Schopenhauer, "The world is my representation"[1]. If that is true for all of us, artists go one step further when creating original works. They imagine parallel realities that either mirror their representation of the world (realism), fulfill perceived expectations (usually for material gain), propose an alternative model (politically motivated), represent a gamesome construction (abstract art) or combine a number of such goals. Some of these creations are cleverer than other, some are more complex, some more attractive and some more subtle but all divulge the artist's system of beliefs. The work under scrutiny here, *figer*, is not different: its architecture, syntax and purported scenario denote a particular view of the world.

Having established that, two questions could be asked: are the traditional roles of the composer and of the audience transformed in the context of what Umberto Eco's defines as

*open works* [2] ? and can a narrative be identified in an "abstract" composition ?

## 2. Form and Structure

### 2.1 Composition Software

*figer* is a computer-assisted composition for computer-generated sounds (fixed media) that can be presented in stereo or eight channel versions. It is also a *manifold composition* [3] or a *class of compositions* whose multiple variants are realized by a computer. In the case of *manifold compositions*, the computer runs a program containing elements of indeterminacy and reads the same data for each run, the only variable being the random number generator's seed. The work was realized with DISSCO, software that unifies composition and sound synthesis into a seamless process [4] for the purpose of realizing such *manifold compositions*.

In DISSCO, written in C++, objects representing the formal units of a composition are arranged in a Directed Graph (DG) tree structure and inherit from an Event class. There are: a unique Top Event corresponding to the entire piece; High Events - main sections; Medium and Low Events - corresponding to more detailed divisions; and Bottom Events in charge of generating synthesized sounds or notes in a score. Each level can spawn any number of Children Events and the same operations are available at all levels of the hierarchy. The user is offered an array of tools that can be used to choose values for assorted parameters and nested link lists that increase considerably the scope and sophistication of such decisions. Applying similar procedures to different time scales insures the coherence of the whole.

DISSCO offers three ways in which Children Events can be created: *Continuum*, *Sweep*, and *Discrete*. *Continuum* distributes values randomly within a designated space and, in *figer*, most attacks and durations are determined this way. *Sweep* creates sequence of attacks and insures that no event starts before the end of the preceding one plus a certain interval (which can be 0). It is used to place such events in succession. *Discrete* assigns attacks and durations for multiple streams of events. It relies on a three dimensional matrix built with the help of two sieves [5].

Deterministic options are available but the software displays a bias toward methods based on probabilities. From choices regarding the macro structure of the piece to those related to minute details in a sound's makeup, the user can introduce elements of randomness that, in turn, support the making of *manifolds*. Depending on the extent to which probabilities were used, each computer run will produce a different distribution of sounds, new sonorities or even multiple formal designs.

Tools that introduce indeterminate aspects in a composition range from flat distributions within a specified range, to functions that control average values, to a triplet of envelopes (functions of time) two representing the lower and the upper dynamic limits of the range, respectively, and the third the probability distribution defined between these limits. Such envelopes are stored in a library and could be drawn by the user who can either select points on a line and drag them to the desired position or specify precise  $(x, y)$  coordinates. They can also be constructed through the *Make Envelope* method which allows the composer to define a range of values for each  $x$  and  $y$  point thus introducing another element of indeterminacy in the process. In this case, the same envelope may take a different shape every time it is used.

## 2.2 Time Proportions

The Top Event in *figer* has eight children or High Events: four main sections, three interludes and a coda. The main sections have the durations of 5, 2, 3, and 1 minutes, the interludes last for 1/3, 1/5 and 1/2 minutes

while the coda has a duration of 1/3 minutes.

The High Events or main sections contain between four and nine streams represented by Medium or Low Events. The two minute main section contains four such streams, the three and the five minute sections consist of five streams each and the one minute High Event presents nine streams. Reminiscent of Cage's square root method or rather of his way of organizing the music based on time intervals and not on pitch, the beginnings and endings of these streams divide the main sections into segments whose lengths are defined by Fibonacci numbers or Golden Sections and reflect the divisions of the Top Event.

## 2.3 Types of Materials

Only three primary types of materials are present in *figer*: points, lines, and chords/textures or simultaneous aggregates. Points are disjointed, isolated, unique occurrences: most of them are loud percussive sounds usually lasting a fraction of a second but some are longer "shrieks" with a more complex internal content.

Lines are either drones persisting for up to two and half minutes or long continuous glissandi between well defined points that may evolve for more than three minutes. Such glissandi are actually single, prolonged sounds produced by applying an envelope to the initial frequency. Frequencies of the pitches forming this uninterrupted "melody" are carefully computed as points where there is a change in envelope's direction and angle.

Chords are pitch aggregates whose components share the same start time. They can form familiar sonorities (such as traditional diatonic or chromatic chords), clusters or complex sound mass textures whose constituent units are indistinguishable. As in the case of points and lines, their particular frequencies could be arbitrary stops either on a continuum, or on equal tempered scales or on just intonation scales (as in the case of the coda section).

The three categories of primary materials could be combined in order to create



intermediary shades. Isolated points form sound mass textures when the density increases beyond a certain level. Lines become sometimes tangled in a complex counterpoint to the point of forming an opaque conglomerate. What might be recognized as “siren sounds” are extended points on the verge of becoming lines and changes of register might dramatically alter the appearance of any primary type. High pitched drones of constant frequency but circulating around the room blur the difference between static drones and dynamic lines. Finally, white noise, used sparingly, is treated as one point but is it actually a drone (line) or a sound mass texture, an agglomeration of random frequencies ?

Simple, straight forward and basic types of materials could become ambiguous when appearing in different circumstances and *figer* takes advantage of such vagueness.

## 2.4 Sound Synthesis

DISSCO generates sounds using LASS, a Library for Additive Sound Synthesis. In an electro-acoustic music environment dominated by *musique concrète* techniques, additive synthesis stands out and it is easily recognizable. When composing *figer* a challenge has been to create sounds that are not quickly identified as such while taking advantage of the main aspect of additive synthesis: strict control over details. As a result, some sonorities recall your typical “additive” sound – especially the lines in the 5 min. section – while others are more difficult to identify.

In the definition of timbre the most important element is the sound's spectral (overtone) content: the number, relative strength, and behavior in time of partials. Sounds in LASS can have an arbitrary number of partials, their amplitude evolution and scaling determined by user defined envelopes. A variety of timbres was produced in *figer* by modifying the harmonic spectra in which partial frequencies are integer multiples of the fundamental. Adding or subtracting a percentage of up to half of the original frequency of each partial (an integer multiple of the fundamental) creates a

distortion of the original spectrum. Since the operation is based on a user specified probability, the result is different for sounds that have the same number of partials and same envelopes even if they were identically scaled originally.

Another precision tool provided by LASS and used in *figer* allows the user to specify the perceived loudness, a subjective sensation distinct from amplitude, and to maintain it constant over the entire frequency domain. Unique in its category, the loudness device renders obsolete the post-production stage during which amplitude overflow is usually managed and makes possible the creation of *manifolds*.

*Modifiers* are another way through which typical additive synthesis sonorities are transformed. They include vibrato (FM), tremolo (AM), glissandi, and transients. All are controlled by the user stipulating the following: a probability of occurrence, the magnitude of the modulation factor (Index in FM synthesis), and its rate. If a six to eight Hertz vibrato or tremolo might be construed as “normal”, asking for much higher rates (50-200 Hz or even more) would create an effect not unlike that produced through the FM synthesis technique even when the carrier remains at a frequency well within the 20 to 4000 Hz. range. In the case of transients - spikes in the frequency or in the amplitude of a sound - the user can also control their width or the number of samples affected. While for acoustic instruments transients occur only in a few fractions of a second at the onset or the end of a sound, in DISSCO they can be placed during the entire duration and are controlled by envelopes.

Spatialization creates two categories of *manifold* versions of *figer*: one for stereo and another for an eight channel array. In both cases, an effort has been made to coordinate the suggested spatial location of sounds with their loudness and the amount of reverberation involved in order to obtain a more “credible” effect.

For the eight channel versions two methods available in DISSCO, *Multipan* and *Polar*, were used. The first enables the composer to assign specific percentages of the sound to each

speaker. *Polar* assumes that the speakers are arranged in a circle. Specifying the angle ( $\theta$ ) and the radius ( $r$ ) places the sound in the room. In the 5 minute section, a spatialization “canon” is created by having the lines follow each other.

### 3. Narrative and Continuity

#### 3.1 The Surface

When first approached, *figer* presents itself as a dramatic, even emotional narrative. There are a number of sound categories that could be associated with realistic imagery: one can recognize a heart beat, sirens warning of an impending adversity, “marching chords” moving in from all directions like soldiers threatening to take over the audience. Then there are these assailing shrieks (or maybe they are desperate ?) and a bombardment of small particles in huge numbers creating anxiety or at least a uncomfortable feeling. Melodic lines become tangled in a disconcerting way but long drones and sustained, somewhat familiar chords bring an element of stability and misleading reassurance. At the same time, the long, very high, piercing sound of the ½ minute interlude only increases the level of uneasiness.

From this perspective, the work seems to present an aggressive, disturbing, “apocalyptic” picture full of surrealist aural images. Similar to paintings born from that aesthetics, it includes recognizable, somewhat familiar elements placed in an incongruent context. The coda, a quote from the traditional repertoire, only enforces this perception.

Things become less clear if one is trying to discern a “story” in this music. It will be difficult to find casual links between various components, a thread that leads in one direction toward a necessary conclusion. Unlike the music of a few centuries ago, there are no themes-characters participating in a logical “plot” supported by a consecrated form (eg. Fugue, Sonata) or by a literary program. Instead, there is an offering of various sound objects that re-occur in a non-linear, ostensibly random succession - the elements of this piece

do not seem to be arranged in a logical, deterministic order. Each new appearance of a previously encountered object is distinct in different degrees although they all can be quickly identified as being diverse incarnations of the three primary types of materials: points, lines, and aggregates.

The continuity of the piece is broken by interludes that introduce new materials disconnected from their surroundings. They fulfill at least two functions: to generate surprise and to encourage the disengagement of the audience from the preceding scene. Inserting not only a totally different type of material but also a contrasting way in which it is presented (static or monotonous-repetitive as opposed to the fast paced, dynamic main sections), the interludes are unexpected and disruptive. They also prevent listeners to become immersed in the musical discourse and invites them to switch from a passive mode of reception to an active one, to adopt a critical attitude - a practice found frequently in Bertolt Brecht's plays.

#### 3.2 The Substance

Encountering a new work, the listener's expectations based on previous experiences are measured against the reality of the yet unexplored object. Innovative features are related to familiar ones in order to construct a new paradigm. Sometimes references to extra-musical practices are necessary in order to accomplish the task.

Certain aspects of *figer* present similarities not only with some musical compositions but also with literary or cinematic oeuvres. In the novels of Alain Robbe-Grillet the description of the same incident occurs repeatedly during the work, each time with changed details (see, for example, “La maison de rendez-vous” [6]). In the famous “statue scene” from the movie “L'Année dernière à Marienbad”, directed by Alain Resnais from a screenplay by Alain Robbe-Grillet [7], we hear five possible explanations of the attitudes displayed by the sculpted couple and their dog. The events in this movie align themselves in an ambiguous

time line open to multiple interpretations and, by the end, it is unclear if a crime has been committed or not.

In a similar way, the main sections of *figer* present the same primary events under different guises, same objects viewed from a number of different perspectives. A change of register, a modified duration, a higher or lower rate of vibrato or the placement in a new context does not affect the essence of such objects. Open to multiple interpretation, the piece does not offer a unique solution or a unique timeline, it does not follow a pre-existent formal pattern, and its complexity adds to the equivocation of the whole. The extensive use of probability, from the higher structural levels down to the finest details in the makeup of each sound, add another layer of incertitude and defeat the attempts to congeal and reduce this piece to a fixed manifestation: *figer* is a *manifold composition* comprising a unlimited number of variants.

There are some obvious similarities between concepts animating *figer* and aleatory music approaches in particular the “Momentform” of Stockhausen [8]. They are regarded here as part of that group of past experiences against which a new artifact is measured.

Music has the advantage over other art forms of being abstract which makes it easier for the composer to invite the listeners to construct their own interpretation. This piece could also be seen as a riddle whose answer lays with the coda and the title. Each individual will forge a distinct representation of the realm proposed by the composer, aware of being a co-author.

Ultimately, *figer* has its own narrative represented by its internal coherence, by the juxtaposition of various materials in an organic conglomerate. Facing this “found object”, the listener is encouraged to take an active role and contribute to the creation of an imaginary world, a parallel reality.

#### 4. World View

Music is not created in a cultural vacuum and most pieces reflect to some extent the

“Zeitgeist”, the spirit of their time. Longer periods also share a dominant way of looking at and understanding reality, a shared world view that influences an individual's representation mentioned by Schopenhauer. This world view is echoed, in the architecture of a work, in the materials used and mostly, in the musical language employed. Tonality, for example, started in the seventeenth century as an orderly hierarchical system governed by causality, in perfect concordance with the rationalism of the Enlightenment, only to dissolve a few centuries later in a mass of ambiguities and delayed resolutions mirroring more complex scientific and less stable social environments.

Composers have the responsibility to recognize such matters and to be aware of the consequences brought about by the artistic choices they make - it is in their power to become active participants in a contemporary reality or, even better, to design a desirable world. Those who deny being aware of such facts and avoid being accountable, are what Herbert Brün called “self appointed morons”. In his definition: “the self-appointed moron, having some sort of intelligence, uses it to avoid thinking about knowledge” [9].

The world of *figer* does not avoid responsibility: it combines a structure based on a rhythmic template that is stable, even rigid in its outlines with surface occurrences that are fortuitous. The rational framework is grounded in simple proportions and is created through deterministic means such as sieves and basic logical (Boolean) operations. At the level of details, accidental happenings are governed by probability functions. However, such random occurrences may exist only within the confines of the larger structure which is driven by causality. Thus, *figer*, becomes a small model of the real world as described by contemporary science: ruled by indeterminacy at the particle level while large bodies obey the laws of classical mechanics. Another way to look at this would be to recognize that no chance events or human actions can contradict the laws of Nature; in the world of *figer* there is no room for miracles.

A relativistic view also characterizes the work. Its sections are presenting the same materials in

different arrangements but without developing them. Same entities are accessed from different angles, all equally valid, none of them privileged. The audience is free to adopt an advantage point of view but there is none implied in the makeup of the piece. In this world there is no one-and-only absolute Truth, a point of reference indicating the center - as in tonal music. In the case of *manifold compositions*, all variants are equal, none is “better” than the next and an exhaustive knowledge of the work - that absolute Truth - can be arrived at only by listening to all potential variants, an impossible task since their number is practically infinite.

DISSCO is a comprehensive or “black box” type of software. Meaning that after planing ahead, the composer feeds in all necessary data, runs the program without intervening in the process and accepts the result unless obvious blunders in the planning or in the preparation of the input are discovered. Since probability has a significant contribution in producing the work, the role of the composer shifts from the author of a unique piece to that of the creator of an entire universe. In order for a software to work on a large scale and in a multitude of heterogeneous situations, generating compelling aural realities, it has to be consistent within itself, to embody a system in which all variables are bound by interconnected constraints. Because DISSCO has this kind of internal coherence (minding the famous critique of Boulez at the address of Schoenberg's music), *figer's* world is plausible, a real life-like parallel world.

## 5. Conclusions

It turns out that an “abstract” work can still deliver a narrative albeit one that each member of the audience constructs independently especially if it belongs to the *open work* category. However, in spite of the variety of representations, the internal structure of the composition provides an abstract scenario, a template that unveils its own rich essence.

A *manifold composition*, *figer* betrays a world view in which structure and chance, destiny and free will, determinism and randomness coexist; it offers a vision of a

relativistic world without privileges in which one has the means to create new parallel realities. The question remaining is: are these multiverses, multiple worlds where variations on the same type of events happen in different guises as postulated by the Many Worlds interpretation of quantum physics or are they, like everything else, under the power of the Hindu Maya - illusions, subjective representations ?

## 6. References

- [1] Arthur Schopenhauer, *The World as Will and Representation*, Dover, 1969.
- [2] Umberto Eco, *Opera Aperta*, Bompiani, 1962.
- [3] Sever Tipei, "Manifold Compositions: Formal Control, Intuition, and the Case for Comprehensive Software", in *Proc. Int'l Computer Music Conference* Copenhagen, 2007, vol. II, pp. 429-436.
- [4] Hans G. Kaper and Sever Tipei, “DISSCO: A Unified Approach to Sound Synthesis and Composition”, in *Proc. Int'l Computer Music Conference*, Barcelona, 2005, pp. 375-378.
- [5] Hans G. Kaper and Sever Tipei, “DISSCO: An Object-oriented System for Music Composition and Sound Design”, in *Proc. Int'l Computer Music Conference*, Berlin, 2000, pp. 340-343.
- [6] Alain Robbe-Grillet, *La maison de rendez-vous*, Paris: Les Éditions de Minuit, 1965
- [7] Alain Robbe-Grillet, *L'Année dernière à Marienbad: ciné-roman*. Paris: Les Éditions de Minuit, 1961
- [8] Stockhausen, Karlheinz. 1963a. "Momentform: Neue Beziehungen zwischen Aufführungsdauer, Werkdauer und Moment" *Texte zur Musik*, vol. 1, pp. 189-210. Cologne: DuMont Schauberg.
- [9] Herbert Brün, “The Premise is That There Be Music”. keynote paper, in *Proc. Int'l Computer Music Conference*, Vancouver, 1985, pp. 1-4.

# The Maxwell Demon: Comprovisation in Ecologically Grounded Creative Practice

Luzilei Aliel <sup>1</sup>\*, Damián Keller, <sup>2</sup> Rogério Costa <sup>1</sup>

<sup>1</sup> ECA – Escola de Comunicação e Artes da Universidade de São Paulo  
Prédio Central 1, Av. Prof. Lúcio Martins Rodrigues, 443 - Butantã, São Paulo - SP, 05508-020

<sup>2</sup> Núcleo Amazônico de Pesquisa Musical, Universidade Federal do Acre  
Rodovia BR 364, Km 04, s/n - Distrito Industrial, Rio Branco - AC, 69920-900

first\_luzaliel@usp.br, second\_dkeller@ccrma.stanford.edu, Third\_rogercos@usp.br

## Abstract

This paper aims to expand the research on ecological synthesis (Keller, 1999) through the inclusion of improvisation practice. We propose a formalization of creative processes in sonic improvisatory-compositional environments (targeting comprovisation), based on ecologically grounded creative practices. Our approach entails the use of socio-ecological models that deal with complex adaptive systems [Sibertin et al., 2011]. We developed a performance/experiment called The Maxwell Demon, as a case study. The observations done during the case study indicate that imitation is an important strategy for creative activities in socio-ecological systems. Improvisation may provide a relevant source of sonic content in ecological environments, enhancing their flexibility without losing consistency.

## 1. Introduction

This work deals with ecologically grounded creative practice targeting both composition and sonic improvisation (comprovisation). We take as reference complex adaptive systems due to the large number of variables embedded in this type of approach. The assessment of responses of complex adaptive systems to dynamically changing scenarios are usually approached from a modeling perspective [Barreiro and Keller, 2010]. Models can be employed to observe socio-ecological dynamics, such as the one proposed by Sibertin-Blanc et al (2011) to determine the qualitative processes in complex

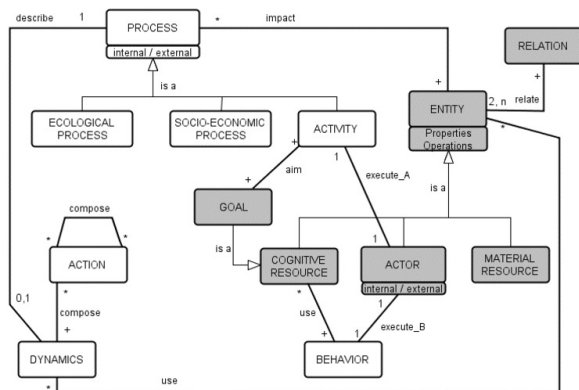
adaptive systems. We discuss the alignments and deviations of this proposal with current eco-compositional and comprovisational practices. To account for the improvisatory elements involved in ecologically grounded creative practice, we expand the eco-compositional approach laid out by Keller (1999) with a Proposal of Modeling of Ecological Synthesis (PMES). Socio-ecological models were developed to design performances/experiments, conducted as case studies. Analytical notes include creative processes, interaction strategies, types of agents, and the production of relevant or disposable resources. We employed, tested and observed the outcomes of the model in a performance/experiment that uses ecologically grounded comprovisation, The Maxwell Demon (TMD). We use a model of adaptive complex systems to formalize the implementation of a proposal for ecological modeling (PESM) using improvisational resources.

## 2. Socio-Ecological Systems

Socio-ecological systems are complex adaptive systems that are characterized by self-organization and distributed control. In socio-ecological systems, social and ecological processes interact at various temporal levels. These reorganizations entail emerging structures and functions. Therefore, models usually encompass multiple agents with diverse and contrasting objectives, being observed at various spatial and temporal levels (Reed, 2008; Pahl-Wostl, 2007; Giampietro 2002). According to Sibertin-Blanc et al. (2011), a socio-

\* Supported by FAPESP

ecological system includes *entities* and *processes*, together with relationships between the entities. The relationships among these three types of entities produce *instances of entities*, which are resources that may appear or be discarded over time, as the state of system changes. Entities are characterized by properties whose values represent the state of an instance<sup>1</sup>. There are two types of relationships among entities: 1. *structural relationships* are associated to the entities by their nature; 2. *non-structural relationships* are created as a result of agents actions. The dynamic character of the socio-ecological system involves processes that evolve toward orderly or disorganised conditions, impacting both the internal and the external processes. Hence, it is necessary to consider the interactions between the socio-ecological system and the environment.



**Figure 1: A model of socio-ecological systems proposed by Sibertin-Blanc et al. (2011).**

### 3. Ecological Synthesis Models (PESM)

Keller's (1999) ecological modeling conceptualizes the creative use of environmental sounds, proposing spatial locations consistent with the sound sources and applying ecologically viable sonic transformations. The ecological approach is based on the premise that all sonic models

1 - This term comes from computer science. The concept of instance corresponds to the existence of an object that shares some characteristics with another individual or object. For example: despite some singularities (type of instruments, technical training, sonic preferences, etc.), all musicians have similar characteristics to other individuals of the same class (such as their musical training encompassing domain-specific knowledge).

should be restricted to ecologically feasible events. Ecological validity is defined by observations of complex interactions occurring in the environment, i.e. by data of agents-objects interactions. The individual's actions on the environment and the influence of the environment on the individual determine a process of pattern formation. This process can be modeled by algorithmic tools [Keller, 1999: p. 23]. Keller (1999) argues that synthesized sounds can expand the compositional resources without compromising the consistency of the sampled events, hence providing unique opportunities for creative action. The author puts forth two techniques to expand the creative possibilities afforded by synthesized sounds in ecologically based scenarios: 1) Generic Physical Models, 2) Control of Meso-Level Granular Sample Sets (see Keller 1999 for a detailed description). The ecological approach places stress on the usage of material resources: 1) actual environments yield resources where the agents are located while carrying out their activities; 2) synthesized environments incorporate resources through digital audio processing; 3) *meta-soundscapes* encompass resources originated in local and remote environments [Ariel; Fornari, 2013]. The social process, in turn, generates phenomena resulting from interactions among human beings. We consider three strategies: 1) *imitation*: the ability to employ mimesis based on perception, analysis and synthesis [Mannis, 2014]; 2) *exploratory activity*: the agent's ability to discover material resources and seek to develop interactions with environment, other agents and *gelassenheit*<sup>2</sup> entities; and 3) *epistemic activity*: knowledge construction in an empirical way, through production of creative material [Keller et al., 2010].

### 4. Comprovisation in PESH

The practice of comprovisation is a recent

2 - (Heidegger, 1966; Koutsomichalis, 2011). They are pure contingents in the environment that can not be quantitatively measured or evaluated: error or indeterminacy. The concept of *gelassenheit* (in a sound path Koutsomichalis, 2011) deals with the ability to describe the particular quality of a sound mass. A distinct set of conditions that only can be experienced subjectively.



## 5. Case Study: The Maxwell Demon: Materials and Methods

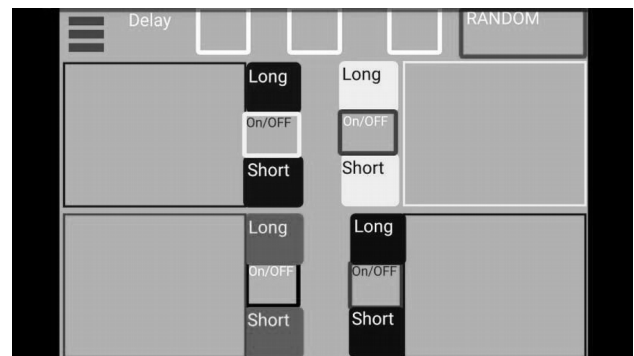
### 5.1 Proposal

The Maxwell's Demon (TMD) is a improvisation inspired by James Clerk Maxwell's 1871 experiment. In this experiment, the Maxwell Demon is an imaginary creature designed to contradict the second law of thermodynamics, the tendency of every system towards entropy. Maxwell's experiment can be represented as a box with a divider placed in the middle, separating it two compartments, left and right. This partition has a door that can be opened and closed by an imaginary being, called Maxwell's Demon. The demon opens the door to allow only the fastest molecules to flow to one side of the chamber. Only the slower molecules flow to the other side, gradually causing one side to warm up, while the other remains cool. Thus entropy is reduced. We use TMD as an artistic metaphor focused on sound (rather than thermodynamics) to simulate an imaginary being - in our case, a stochastic algorithm - that seeks to control the sonic outcome to increase or reduce its entropy. Conceptually, we treat stochastic algorithms as *Gelassenheit* entities [Heidegger, 1966; Koutsomichalis, 2011]. A *Gelassenheit* entity has an "independence" in time and space, its dynamics are established by stochastic processes.

### 5.2 Design/Implementation - Guidelines Plan and Contingency Plan

*Materials/Equipments:* We designed of a tool capable of producing sounds: 1) easy to manipulate; 2) accessible to all agents (through deployment on mobile platforms based on Android and IOS systems). All participants (musicians/non-musicians) were given a mobile phone with a *Pure Data* (PD) [Puckette, 1997] patch adapted for *MobMuPlat* [Iglesia, 2016]. The mobile screen features four rectangles that act as controllers of additive synthesis oscillators. Simultaneous control of up to four banks of oscillators is possible. Four FM synthesis oscillators feature control parameters

for frequency, duration and delay. Random processes are controlled by tapping the phone's screen and can be turned on and off at any time. The frequencies vary from 220 Hertz to 1320 Hertz. Frequency increments are associated with gestures from left to right. Beside each rectangle there are three oscillators switches (on/off buttons) and two envelope controllers. One with a condition of attack, decay, sustain and release short and another with envelopes in longer conditions. The left button at the top of the screen controls a stochastic algorithm connected to all the oscillators frequencies. This triggers random changes in each oscillator. The remaining three buttons control delay processing of the sound material. From left to right, there are fixed delay rates of 150 ms., 300 ms. and 750 ms.



**Figure 3. Interface of the The Maxwell Demon mobile patch.**

The computer running the automated algorithm is connected to four loudspeakers. The loudspeakers were placed at the four corners of the studio where the performance/experiment<sup>3</sup> took place. The agents moved around the perimeter of the environment. We developed a patch (PD) that runs on a desktop computer with similar sonic features as the algorithms for the mobiles. Rather than being controlled by the participants, stochastic automated processes determine when and how sound events will occur. With the touch of the green start button, the entire performance/sound experiment occurs in an automated way. At the ringing of the bell begins the artistic narrative finalizes.

*Sonic materials* - We use an emulated bell (based on FM) that plays at the beginning of the

<sup>3</sup> - Experiments developed to assess artistic results qualitatively or quantitatively.



performance/experiment and ends it when it is heard again. This sound is triggered by a PD patch. The entire performance takes seven minutes. All sound content that occurs between the ringing of the bells are contingencies resulting from interactions and sound discovery.

*Location and participants* - The experiment was performed in a small-sized studio, approximately 10x07 meters. Having Maxwell's procedures as inspiration, we use two types of agents, those with traditional knowledge of music and those with little or no knowledge. Participants included five musicians and two non-musicians. All participants had college education. Non-musicians possessed familiarity with basic musical concepts but had no formal training. Among the non-musicians, there were three women - ages 25, 32 and 35. The musicians were four men, ages 26 to 58 and one woman (22 years old). Music training and previous musical experience varied from 10 to 30 years.

*Assessment* - Data was collected through interviews with the participants, on site observations and analysis of audio and video recordings. The objective was to assess the interactions among the agents and how the resources are used, and to expose the behavioral effects of the socio-ecological system, including: 1) initial state of the entities; 2) internal process dynamics; 3) impact of external processes on entities. The integrated assessment of the scenarios is performed by analyzing the values of the indicators derived from the performance/experience process and from the final state of the entities.

*Procedures* - Procedurally, TMD is a improvisation [Aliel et al, 2016]. Hence, we create a guideline plan and contingency plan to outline which events will be designed and tested (composed resources) and leave other aspects to occur in an unpredictable manner (improvised resources). The experiment adopts a socio-ecological approach where agents and algorithms relate sonically. The performance/experience is guided by the definition of scenarios [Sibertin-Blanc et al. 2011]. A performance/experiment is defined as a scenario featuring a free improvisation with

cell phones lasting approximately seven minutes. The only guidelines given to the agents are: "move through space and use mobiles to produce sounds". After ringing the first bell, the computer starts the pre-set parameters and selects the number of oscillators, pitches, dynamics, durations and delay processing.

## 6. Results of The Maxwell Demon Case Study

*Contingencies* - We consider interactions and behaviors leading to sound discovery as sources of contingencies. Much of the material produced in the TMD sessions was rarely repeated, providing conditions of low sonic pregnancy. The sounds produced by the algorithm were dynamically related to the actions of the agents. Nevertheless, a *Gelassenheit* entity produces sound content that may or may not be imitated by or contrasted with the outcomes produced directly by the agents. In this sense, this guideline seems to be analogous to Maxwell's imaginary entity.

*Behaviors* - The agents explored various material resources contained in the environment, generating new forms of interaction. Their mobility - in addition to the low dynamic range of the sounds produced by the mobile devices - provided a sonically dynamic and highly concentrated environment in which the focus of attention changed constantly.

*Musical Expertise* - Although there were disparities of musical knowledge among the agents, all showed similar technical ability while trying to produce sounds. We observed similarities in the three performances/experiments, involving intense agent interactions. The exchanges encompassed: algorithms x agents, agents x agents and agents x environmental resources.

*Imitation* - We observed that a large part of the interaction process was driven by imitation (a strategy pointed out by Mannis, 2014). Choices of processing types and dynamic changes of parameters were predominant. In general, there was a prevalence of imitation of processes suggested by musicians, but contents managed

by non-musicians were also present.

### 6.1 Implications of the Artistic Outcomes

*Socio-ecological systems* - By introducing ecologically grounded sound synthesis strategies in improvised contexts, we target unique conditions for each performance. The performance/experiment TMD thus integrates the electronic sound structures into socially dynamic forms of interaction, approaching the complexity of biophonic ecologies. When introducing an ecological synthesis model within a socio-ecological system, the sonic outcomes may follow deterministic rules (composition) or adopt strategies based on contingencies (improvisation). By adding resources such as sound synthesis and audio processing based upon the actions of agents within a sound ecology, we introduce material and cognitive resources that are characteristic of socio-ecological systems. Aligned with the improvisation methodologies, the PESM features material resources that can be used as guidelines or contingency plans.

*Exploratory behavior.* The materials collected in previous case studies [Aliel et al, 2015a and 2015b] indicate that the generation, maintenance and disappearance of sound material in improvisatory ecological contexts is linked to imitative cognitive exchanges [Mannis, 2014], exploratory or epistemic activities [Keller et al., 2010] targeting an increased knowledge of material resources by the agents. These exchanges may be limited by the adaptability of the behaviors indicated by Sibertin-Blanc et al. (2011) (see introduction).

### 7. Final Considerations

Taking into account the socio-ecological system prerequisites, we elaborated a Proposal for Ecological Synthesis Model (PESM) based on Keller's (1999) notes. While expanding the range of ecocompositional applications, we highlighted the possibility of including improvisation in sonic ecologies. The PESM was applied in a case study called The Maxwell Demon. Three sessions were carried out with the objective of collecting information on the agents behaviors, the sonic resources and the

technological support. These observations yielded proposals applicable to ecologically grounded artistic works that target both musicians and non-musicians. Future studies may evaluate the actions of the agents while attempting to deal with unpredictable conditions, such as those found in alternative and outdoor venues.

### References

- [1] Aliel, L., and Fornari, J. Projeto Destino Pirilampo: Um Estudo sobre a Composição de Meta- Soundscapes em Música Ubíqua. *Música Hodie*, v.14, p. 105- 121. 2014.
- [2] Aliel, L. and Costa, R. . 'Sons sem Sino' - Abordagens Comprovisatórias em Instalações Sonoras. In: *IV Jornada Discente - ECA, 2016*, São Paulo. IV Jornada Discente - ECA, 2016.
- [3] Aliel, L.; Keller, D. ; Costa, R. ; Melo, M. T. S.; and Pinheiro da Silva, F.; Santos, L. A.. Eco-cognitive Improvisational practice: Two case studies. In: *VI Ubimus, 2015, Vaxjo*. Ubiquitous Musical Ecologie, 2015a.
- [4] Aliel, L.; Keller, D. and Costa, R.. Comprovisação Abordagens Desde a Heurística Estética em Ecocomposição. In: *SBCM - XV Simpósio Brasileiro de Computação Musical, 2015*, Campinas/SP. SBCM - XV Simpósio Brasileiro de Computação Musical, 2015b.
- [5] Aliel, L.; Keller, D.; and Costa, R.; Perspectivas Teóricas para a Análise das Práticas Criativas Ecocognitivas. *Per Musi* (UFMG), 2016.
- [6] Barreiro, D. L. and Keller, D. Composing with sonic models: fundamentals and electroacoustic applications. In D. Keller & R. Budasz (eds.), *Criação musical e tecnologias: teoria e prática interdisciplinar, Vol. Criação Musical e Tecnologias: Teoria e Prática Interdisciplinar* (pp. 97-126). 2010.
- [7] Cardew, C. A Scratch Orchestra: Draft Constitution. *The Musical Times* 110 (1516), 617-619. 125th Anniversary Issue. 1969.
- [8] Chen, P. and Pin-Shanhan, P., The Entity-Relationship Model - Toward a Unified View of Data. *ACM Transactions on Database Systems*

1 (1): 9–36. 1976

[9] Curran, A. and Teitelbaum, R. *Musica Elettronica Viva. For MEV, program notes.* New York: The Knitting Factory. <http://www.alvincurran.com/writings/mev.html>. 1989.

[10] Di Scipio, A. The place and meaning of computing in a sound relationship of man, machines, and environment. In A. Georgaki & G. Kouroupetroglou (eds.), *Proceedings of the ICMC|SMC|2014*. Athens, Greece: ICMA. 2014.

[11] Ferraz, S. and Keller, D. Preliminary proposal of the MDF model of collective creation. *Cadernos de Informática* 8 (2), 57-67. 2014.

[12] Iglesia, D. The Mobility is the Message: the Development and Uses of Mob Mu plat. In: *PdCon16.NYC*. 2016. <http://www.danieliglesia.com/mobmuplat/IglesiaMobMuPlatPaper.pdf>

[13] Giampietro, M, Complexity and Scales: The Challenge for Integrated Assessment. *Integrated Assessment*, 3 (2–3), 247–265. 2002.

[14] Keller, D. touch'n'go: Ecological Models in Composition. Master of Fine Arts Thesis, Burnaby, BC: *Simon Fraser University*. <http://www.sfu.ca/sonicstudio/srs/EcoModelsComposition/Title.html>. 1999.

[15] Keller, D. Sonic Ecologies. In A. R. Brown (ed.), *Sound Musicianship: Understanding the Crafts of Music* (pp. 213-227). Newcastle upon Tyne, UK: Cambridge Scholars Publishing. (ISBN: 978-1-4438-3912-9.) 2012.

[16] Keller, D., Barreiro, D. L., Queiroz, M. and Pimenta, M. S. Anchoring in ubiquitous musical activities. In: *Proceedings of the International Computer Music Conference* (pp. 319-326). Ann Arbor, MI: MPublishing, University of Michigan Library. 2010.

[17] Keller, D., Lazzarini, V. and Pimenta, M. S.

Ubimus through the lens of creativity theories. In D. Keller, V. Lazzarini & M. S. Pimenta (ed.), *Ubiquitous Music* (pp. 3-23). Berlin and Heidelberg: Springer International Publishing. (ISBN: 978-3-319-11151-3.) 2014.

[18] Heidegger, M. *Gelassenheit*: HarperCollins. 1966.

[19] Koutsomichalis, M Site Specific Live Electronic Music: A Sound Artist's Perspective. *Proceedings of the Electroacoustic Music Studies Conference, Sforzando!*, New York. <[http://www.emsnetwork.org/IMG/pdf\\_EMS11\\_Koutsomichalis.pdf](http://www.emsnetwork.org/IMG/pdf_EMS11_Koutsomichalis.pdf)>. 2011.

[20] Latour, B. Reassembling the social: An introduction to Actor-Network Theory. Oxford, UK: *Oxford University Press*. 2005.

[21] Mannis, J. A. Processos Cognitivos de Percepção, Análise e Síntese Atuando no Processo Criativo Mimesis de Mimesis. In: *Encontro Nacional de Composição Musical de Londrina - EnCom 2014*. 2014

[22] Pahl-Wostl, C., Transitions towards adaptive management of water facing climate and global change. *Water Resources Management*, 21: 49-62. 2007.

[23] Reed, M. S., Stakeholder participation for environmental management: A literature review. *Biological Conservation*, 141: 2417-2431. 2008.

[24] Rhodes, M. An analysis of creativity. *The Phi Delta Kappan* 42, 305-311. 1961.

[25] Rotmans, J. and Asselt, M., Integrated assessment: A growing child on its way to maturity. *Climatic Change* 34 (3) 327. 1996.

[26] Sibertin-Blanc, C., Théron, O. and Monteil, C., Mazzega, P.: Formal modeling of social- ecological systems. In: *European Social Simulation Association Conference, Cemagref* 2011.

# Web Orchestra Studio: a real-time interactive platform for music and education

Juliano Kestenberg<sup>1\*</sup>, Vitor Rolla<sup>1\*</sup>, Djalma Lúcio<sup>1</sup>, Luiz Velho<sup>1</sup>

<sup>1</sup>VISGRAF Laboratory

Instituto Nacional de Matemática Pura e Aplicada – IMPA

Estrada Dona Castorina 110, Jardim Botânico – 22460-320 – Rio de Janeiro, RJ – Brasil

julianok@impa.br, vitorgr@impa.br, dlucio@impa.br, lvelho@impa.br

## Abstract

In this paper, we introduce the Web Orchestra Studio, a set of applications which enables the development of musical concerts for laptop orchestras. We offer an open-ended platform for collective artistic experimentation which can be utilized by experts or non-initiated students. In order to instance some of the platform features, we present a case study describing our participation in the Math Festival activities with the workshop Music, Mathematics and Computers. Fundamentally, with this work we intend to leverage academic debate concerning the interdisciplinary fields of music, computer science and education.

## 1. Introduction

In the past decade the first laptop orchestras appeared. Generally, such ensembles are formed by academic professionals who have background knowledge in audio programming and music. Usually, the orchestras are associated with a post-graduate program within an academic context. Therefore, they are considered a fruitful environment for conducting interdisciplinary research in technology and music.

In this paper, we introduce a web platform called Web Orchestra Studio. This platform was utilized to build a workshop experience tailored to children in the age range of 5-12 years old. The event took place in Rio de Janeiro in April 2017, as part of the Math Festival activities.

This paper is organized as follows. In the next section, we present the most well-documented laptop orchestras. In Section 3, we describe the Web Orchestra Studio, a set of applications that make it possible to prepare a concert and build

a laptop orchestra. In Section 4, we report the workshop Music, Mathematics and Computers – presented during the Math Festival – as a case study of our platform. Finally, we expose conclusions and future work.

## 2. Related Work

In this section, the most well-documented laptop orchestras are presented chronologically.

The first laptop orchestra was founded in 2005 at Princeton University. Its activities include presentation of basic programming concepts, individual and group assignments presentation, and rehearsal as an ensemble. PLOrk, or the Princeton Laptop Orchestra [1], uses a homogeneous collection of six-driver hemispherical loudspeakers for instrument-like acoustic dispersion.

The Carnegie Mellon Laptop Orchestra [5] (CMLO) is part of a course in computer music systems and information processing at Carnegie Mellon University. The students learn techniques for audio and MIDI [13] programming, real-time synchronization and scheduling, and music representation. At the end of the course, they must present a piece of music which is performed by the orchestra.

The Milwaukee Laptop Orchestra [7] (MiLO) grounds its practice in free improvisation. The orchestra is based on the NRCI (Networked Resources for Collaborative Improvisation) software, which was developed in Milwaukee University. The NRCI is a suite of Pure Data [3] tools. Instrumental performances and video projections are common practice among the members.

The Stanford Laptop Orchestra [8] (SLOrk) is a large-scale ensemble that explores cutting-

\*Supported by CAPES.

Name	Software	Seats	Since
Princeton Laptop Orchestra (Plork) [1]	Chuck [2], Pure Data [3] or SuperCollider [4]	15	2005
Carnegie Mellon Laptop Orchestra (CMLO) [5]	Serpent [6]	10+	2006
Milwaukee Laptop Orchestra (Milo) [7]	Pure Data	8+	2007
Stanford Laptop Orchestra (Slork) [8]	Chuck in general	20+	2008
Linux Laptop Orchestra (L2ork) at Virginia Tech [9]	Pure Data	15+	2009
The Machine Orchestra at CalArts [10]	Chuck in general	12+	2010
Birmingham Ensemble for Electroacoustic Research (BEER) [11]	SuperCollider	3-5	2011
Cybernetic Orchestra at McMaster University [12]	Chuck, Pure Data or SuperCollider	10+	2012
Web Orchestra Studio	HTML5 + CSS + JavaScript + Python	5+	2017

**Table 1: Laptop Orchestras**

edge technology in music. It provides a platform for research in instrument and sound design, as well as in music composition and performance. The orchestra also offers a classroom environment which combines music, technology, and live coding [14, 15] performance.

The Linux Laptop Orchestra [9] (L2ork) at Virginia Tech relies mainly on Pure Data for audio, video, and graphics processing. They use Nintendo Wiimotes and built-in laptop input devices (e.g. keyboard, track-pad, webcam) as instruments. They also utilize external sound-card combined with custom-built hemispherical speakers for audio output. The ensemble infrastructure currently supports up to 15 fully networked performers.

The Machine Orchestra [10] at the California Institute of the Arts is a mixed ensemble of humans and robotic performers. Its pedagogical focus is to provide the necessary knowledge to create a robotic instrument or to control the set of robotic instruments previously built for the orchestra.

The pedagogical nature of an orchestra is explored by David Ogborn [12] and Scott Wilson et al. [11]. Ogborn presents the laptop orchestra [12] from University of McMaster, Canada. Wilson depicts the Birmingham music group of electroacoustic research [12], United Kingdom.

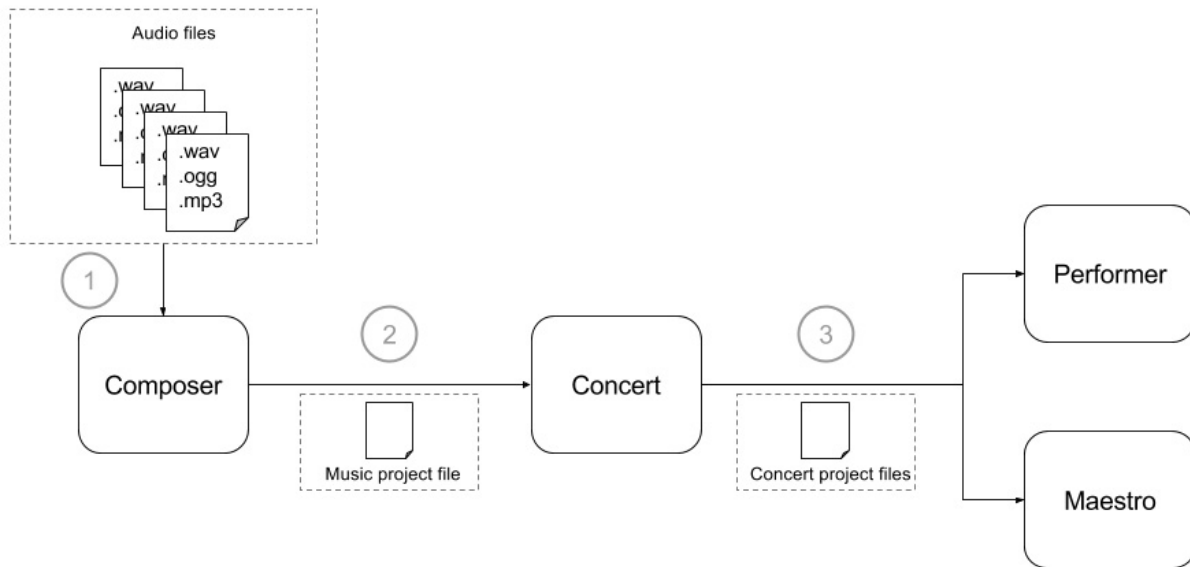
Both explore the participatory aspects involved in a group of musicians and their laptops, promoting shared responsibility in the development of new pieces, and inclusive atmosphere based on peer learning.

The motivations for starting a laptop orchestra, both in musical and cultural terms, and its aesthetic and technical issues can be verified in [16]. An excellent discussion on composing for laptop orchestras is available in [17]. Different strategies for sound design, conduction roles and improvisation are also addressed in the aforementioned paper.

### 3. Web Orchestra Studio

Web Orchestra Studio (WOS) is a set of applications that enables the development of musical concerts for laptop orchestras. The platform comprises the following elements: Composer, Concert, Maestro, Performer and Server. All applications were developed in HTML5, CSS and JavaScript, except the Server – developed in Python language. The communication between the different laptops occurs through websockets [18]. WOS allows one to create a complete concert, from its conception to its deployment in a number of laptops, making them ready for presentation.

The concert creation and deployment process



**Figure 1: Pipeline for musical concert creation and deployment**

is accomplished by following the pipeline depicted in Figure 1:

1. The user composing a musical piece selects the audio files that will be utilized in the composition;
2. In order to define which audio files will be copied to each instrument, a music project file is created using Composer application;
3. Finally, the Concert application generates the concert project files that shall be deployed in the Performer and Maestro applications.

Each application is detailed below:

### 3.1. Composer

The Composer application is responsible for song creation. It stores audio samples in the following formats: WAV, Ogg and MP3. Each song can be played by up to 20 instruments. An instrument can be a computer, a smartphone or a tablet. Each one can play several samples at once.

### 3.2. Concert

The Concert application determines the order in which each music will be played. Then, it forwards the concert project files to the laptops running Maestro and Performer applications.

### 3.3. Performer

The Performer application loads the project created in Concert so that each music can be

played as defined in Composer. Each instrument configured in Composer is associated with a single Performer.

### 3.4. Maestro

The Maestro application loads the project created in Concert and controls which music will be played in the Performer applications. This component interface allows the conductor to see which instrument is being played and how long it will take for its sound to be ceased. It is also possible to interrupt the sound that is being played in the Performer applications anytime.

### 3.5. Server

The Server application is responsible for the communication between Performer and Maestro. Commands sent by Maestro are received by Server and forwarded to every Performer in the network. Reversibly, Performer contacts Maestro through Server. The Server operates with real-time communication.

## 4. Case Study

In this section, we present a case study which illustrates a fraction of our platform features. The workshop Music, Mathematics and Computers has been the selected event. It took place in Rio de Janeiro in April 2017 as part of the Math Festival activities.



**Figure 2: Laptop orchestra performers making eye contact with the conductor**

The workshop main goal was to sensitize students and the general public to the beauty of the relationships between music, mathematics and computers. In order to achieve this goal, we have explored several connections between musical structures, mathematical concepts and computer implementation methods. Our challenge was to provide a rousing atmosphere that could inspire children and teenagers in their learning paths.

The workshop was divided in two main parts. First, the audience was introduced to basic concepts presented along three sections: Introduction to sound and music; Music and Math; and Music with the computer. This theoretical part lasted about 30min. Subsequently, our laptop orchestra was set up, with volunteers – drafted from the audience – playing musical pieces which were previously chosen by our team. The second part took about 20min. Our workshop was given during the four days of the Math Festival. We had a total number of 12 sessions.

#### 4.1. Setup

Our laptop orchestra comprised five computers used by performers and one destined to the conductor. Each performer station had the fol-

lowing components:

1. an HP laptop running Linux Ubuntu 16.04 operating system;
2. a small desk on which the computer rested;
3. a mono-directional speaker for sound amplification;
4. a cushion on which the performer sat.

The five stations were arranged so that the ensemble drew an imaginary semi-circumference on stage, with performers facing the conductor, whose laptop rested on a transparent pulpit. A wired audio network connected computers with a multi-channel audio interface.

#### 4.2. Preparation

After introducing the fundamental concepts, we invited people from the audience to play the computer meta-instruments. Mostly, their ages ranged from 5 to 12 years old.

As the players took seats, we explained the activity dynamics. Keyboard space bar was the only key which would make the computer deliver sound. Performers were asked to make eye contact with the conductor so they could play according to his manual gestures (Figure 2). They

were also made aware that each musical piece would have its corresponding background color on the laptop screen.

### 4.3. Action

Four were the musical pieces performed by our young orchestras:

1. C Major arpeggio;
2. Reproduction of a five-note melody extracted from the movie “Close Encounters of the Third Kind”, directed by Steven Spielberg and released in 1977;
3. Interpretation of *Pontos de Lagrange*, composed by Brazilian computer music researcher Marcelo Cicconet;
4. Interpretation of *Cadência Universal*, also composed by Marcelo Cicconet.

C Major arpeggio served as a warming-up in which performers had the chance to experience the system responsiveness for the first time. Players took turns and delivered a single note, either Dó, Mi or Sol.

In order to reinforce the concepts of rhythm and musical tempo, which had been presented previously, the five-note movie melody was reproduced twice. First, with a slow tempo induced by the conductor. Then, with a fast tempo and notes with a shorter duration.

Both musics composed by Marcelo Cicconet had the purpose of exercising improvisation. Each performer delivered a melodic instrument sample, ranging from piano to electric guitar, to saxophone. The samples were meticulously prepared so they could sound harmonious in multiple combinations.

At first, the conductor was a member of our system conception team. Towards the end of each performance we invited children from the audience to conduct the orchestra. That was a memorable experience as it granted us with the opportunity to observe the system running without our direct control.

### 4.4. Feedback

After each session, Math and Physics teachers approached our team asking for extra information regarding the concepts which had been

introduced. Notably, the teachers expressed their interest in reproducing the workshop in their local learning environments. Hence, they asked for supportive digital material and technical guidance. Finally, children were delighted for having had the opportunity to take part in our laptop orchestra (Figure 3). They shared their thoughts on how to improve the overall performance.



**Figure 3: A laptop orchestra portrayed in the Math Festival**

## 5. Conclusions and Future Work

Laptop orchestras constitute a generous learning environment to expand the overlapping fields of music, computer science, and live performance. With Web Orchestra Studio we offer an open-ended platform for collective artistic experimentation which can be utilized in different instruction levels.

The workshop described in the previous section represents an instantiation of the Web Orchestra Studio. Particularly, we intended to aid children learning quests by awakening playful ways of experiencing mathematics through music and computers.

Our project next steps include the following tasks: (i) make the software suite available for free download; (ii) conclude the project web portal; and (iii) encourage the utilization of the platform as a tool for music and math education.



Ultimately, we believe in the power of combining computers capabilities with the uniqueness of human responses. Web Orchestra Studio is simply one of the countless ways to explore new education paradigms through computer-mediated technology.

## References

- [1] Ge Wang, Dan Trueman, Scott Smallwood, and Perry R Cook. The laptop orchestra as classroom. *Computer Music Journal*, 32(1):26–37, 2008.
- [2] Ge Wang, Perry R Cook, et al. Chuck: A concurrent, on-the-fly, audio programming language. In *Proceedings of the 2003 International Computer Music Conference*, 2003.
- [3] Miller Puckette et al. Pure data: another integrated computer music environment. *Proceedings of the second inter-college computer music concerts*, pages 37–41, 1996.
- [4] James McCartney. Supercollider: a new real time synthesis language. In *Proceedings of the 1996 International Computer Music Conference*, 1996.
- [5] Roger Dannenberg, Sofia Cavaco, and B. Aygun J. Baek E. Barndollar D. Duterte J. Grafton R. Hunter C. Jackson U. Kurokawa D. Makuck T. Mierzejewski M. Rivera D. Torres A. Yu E. Ang, I. Avramovic. The carnegie mellon laptop orchestra. In *Proceedings of the 2007 International Computer Music Conference, vol II*, pages 340–343. The International Computer Music Association, 2007.
- [6] Roger B Dannenberg. A language for interactive audio applications. In *Proceedings of the 2002 International Computer Music Conference*, pages 509–515, 2002.
- [7] Christopher Burns and Greg Surges. Nrci: Software tools for laptop ensemble. In *Proceedings of the 2008 International Computer Music Conference*, 2008.
- [8] Ge Wang, Nicholas J Bryan, Jieun Oh, and Robert Hamilton. Stanford laptop orchestra (slork). In *Proceedings of the 2009 International Computer Music Conference*, 2009.
- [9] Ivika Bukvic, Thomas Martin, Eric Standley, and Michael Matthews. Introducing l2ork: Linux laptop orchestra. In *International Conference on New Interfaces for Musical Expression*, pages 170–173, 2010.
- [10] Ajay Kapur, Michael Darling, Dimitri Diakopoulos, Jim W Murphy, Jordan Hochbaum, Owen Vallis, and Curtis Bahn. The machine orchestra: An ensemble of human laptop performers and robotic musical instruments. *Computer Music Journal*, 35(4):49–63, 2011.
- [11] Scott Wilson, Norah Lorway, Rosalyn Coull, Konstantinos Vasilakos, and Tim Moyers. Free as in beer: Some explorations into structured improvisation using networked live-coding systems. *Computer Music Journal*, 38(1):54–64, 2014.
- [12] David Ogborn. Live coding in a scalable, participatory laptop orchestra. *Computer Music Journal*, 38(1):17–30, 2014.
- [13] Gareth Loy. Musicians make a standard: the midi phenomenon. *Computer Music Journal*, 9(4):8–26, 1985.
- [14] Nick Collins, Alex McLean, Julian Rohrerhuber, and Adrian Ward. Live coding in laptop performance. *Org. Sound*, 8(3):321–330, December 2003.
- [15] Marc J Rubin. The effectiveness of live-coding to teach introductory programming. In *Proceeding of the 44th ACM technical symposium on Computer science education*, pages 651–656. ACM, 2013.
- [16] Dan Trueman. Why a laptop orchestra? *Organised Sound*, 12(02):171–179, 2007.
- [17] Scott Smallwood, Dan Trueman, Perry R Cook, and Ge Wang. Composing for laptop orchestra. *Computer Music Journal*, 32(1):9–25, 2008.
- [18] I. Fette and A. Melnikov. The websocket protocol. RFC 6455, RFC Editor, December 2011.

# Posters



# A tool for the musical education of Deaf people

Erivan Gonçalves Duarte<sup>1\*</sup>, Tiago Fernandes Tavares<sup>1</sup>

<sup>1</sup>Faculdade de Engenharia Elétrica e Computação – Universidade Estadual de Campinas  
Cidade Universitária “Zeferino Vaz” – 13 083-970 Campinas, SP

## Abstract

Hearing impaired people are often neglected musical education. This happens because musical activities are often focused on the development of the sense of hearing. In this work we propose the use of visual and haptic cues for musical education. These cues are generated from the real-time mapping of audio features extracted from a microphone stream. These mappings were incorporated into a mobile app and used in music lessons offered to a hearing impaired community. As a result of these lessons, participants developed the abilities of perceiving and producing rhythms, playing virtual instruments and participating in a collective musical practice.

## 1. Introduction

This work analyzes the use of electronic and computational resources for the inclusion of Deaf<sup>1</sup> students in music courses. Such an inclusion has often been perceived as impossible both by the Deaf Community, teachers, professionals and academics [1]. This idea has been perpetuating for many years, and there has been little effort on the development of musical activities aimed at the Deaf community [2].

In Brazil, music teaching in primary schools became compulsory in 2008 (Law 11768, August 18, 2008) [3]. This brought a new challenge to music educators: the inclusion of students with disabilities, especially in regards to the Deaf.

As Sá [4] points out, technological resources can be used by the Deaf to access music. This opens a new perspective in the possibilities for communication and connection. As a results, technological resources can contribute to make

the Deaf more participatory in society. The development of a technological device to enable the musical education of the Deaf can rely on the idea that musical experiences are not an exclusively auditory, but also social and multisensorial.

## 2. Related work

There are many technology-based mapping systems that produce visual and haptic cues from musical information. Some of them target general audiences like Mitroo [5], Smith [6], McLeod & Wyvill [7], Karam et al. [8] while others, are specifically designed for Deaf users like Kerwins [9], Luiz [10], Nanayakkara [11] and Jack et al [12].

## 3. Proposed approach

We propose a contribution to the musical education of Deaf people comprising both a mobile application and educational remarks that can guide their use within music lessons. The development of the application and their educational counterparts was performed in music workshops in which we gathered feedback from the contact with Deaf people. We developed different interfaces to work each musical skill.

The first interface, extracts musical elements from an audio signal by transforming them into visual and vibrational elements, was aimed at facilitating musical practices comprising the production of sounds at different levels of intensity and frequency. The interface was used to observe and classify the sound production of percussive instruments at different levels of intensity, for the perception and production of rhythm at different *tempi*, for the development of group musical activities in games of production, observation and imitation of rhythms, for the production of vocal sounds in different levels of intensity and pitch,

\*Supported by CNPq.

<sup>1</sup>We stress the term Deaf (capitalized) as a recognition of the identity experienced by the Deaf subjects, their linguistic and social values

for the musical dictation of heights (bass and treble) and, finally, for activities of musical creation using voice and percussion instruments.

Interface 2, allowed the reproduction of musical rhythms, the accomplishment of collective musical practice and facilitated the interaction between the Deaf and the listeners in musical activities. By touching the finger on the device screen the user trigger haptic, visual and sonic responses related to each part of virtual drum. The interface was used in activities comprising the differentiation of the parts of the virtual drum kit according to its haptic response, the reproduction of musical rhythms at different speeds, the execution of rhythms written on a visual score and the participation in a collective musical practice.

Interface 3, allowed participants to differentiate the pitch of musical notes, play small melodic fragments, participate in collective musical practices and facilitated the interaction between the Deaf and the listeners in musical activities. This interface displays a virtual piano, by touching the finger on the screen of the device, the user receives a visual and haptic response, in addition to a sound emission for each note of the keyboard. The interface was used in educational activities involving the identification of musical notes, the reproduction of melodies from a score of colors, the execution of melodies in different *tempi* and the participation in collective musical practices.

#### 4. Conclusion

Our proposal brings forward the possibility of using common techniques derived from traditional musical education, which facilitates its use by music instructors. We emphasize the possibility of developing musical activities for the integration between Deaf and hearing people, as all the interaction using the proposed tool relies on communication through sound. This poses an interesting road for future work.

#### References

- [1] Regina. Finck. *Ensinando música ao aluno surdo: perspectivas para ação pedagógica inclusiva*. PhD thesis, Universidade Federal do Rio Grande do Sul, 2009.
- [2] A.L.C. Cruz. Music for the deaf: a qualitative approach. In L. D. LABBO and S. L. FIELD, editors, *Conference Proceedings of the Qualitative Interest Group*, 2007.
- [3] Decreto n186 de 09 de jul. de 2008. aprova o texto da convenção sobre os direitos das pessoas com deficiência e de seu protocolo facultativo. Diário Oficial–República Federativa do Brasil, Brasília, em 11 de jul, 2008, 2008. Assinado em Nova Iorque, em 30 de março de 2007.
- [4] N. R. L. Sá. *Cultura, Os Surdos, A Música e a Educação*. Edições paulinas. São Paulo, 2007.
- [5] J. B. Mitroo, N. Herman, and N. I. Badler. Movies from music: Visualizing musical compositions. *Siggraph Comput*, 13:218–225, 1979.
- [6] S. SMITH and G. WILLIAMS. A visualiza-tion of music. *Phoenix, Arizona, United States*, pages 499–503, 1997.
- [7] P. McLeod and G. Wyvill. Visualization of musical pitch. *Paper presented at the Proceedings of the Computer Graphics International IEEE. Otago*, 2003.
- [8] M. Karam, G. Nespoli, F. Russo, and D. I. Fels. Modelling perceptual elements of music in a vibrotactile display for deaf users: A field study. In *Proc. 2nd International Conferences on Advances in Computer-Human Interactions*,, pages 249–254, 2009.
- [9] Shane Kerwins. *Vibrato*, 2005. Access date: 16 feb. 2016.
- [10] Luiz T. R. B. *O Uso de Softwares para Estimulação da Percepção do Surdo Frente aos Parâmetros de Velocidade do Ritmo: Proposta de Utilização do Bpm Counter e do Vpm Counter no Programa de Atividades Rítmicas Adaptadas a Pessoas Surdas*. PhD thesis, Universidade Estadual de Campinas, 2008.
- [11] Surunga Chandima NANAYAKKARA. *Enhancing Musical Experience for the Hearing-Impaired using visual and haptics displays*. PhD thesis, Unversidade Nacional de Singapura, Singapura, 2009.
- [12] R. Jack, A. McPherson, and T. Stockman. *Designing tactile musical devices with and for deaf users: A case study*, 2015.

# An Algorithm for Guiding Expectation in Free Improvisational Solo Performances

José Fornari, Stéphan Schaub

Núcleo Interdisciplinar de Comunicação Sonora (NICS) – Universidade Estadual de Campinas  
Rua da Reitoria, 165 – 13083-900 Campinas, SP

[tutifornari@gmail.com](mailto:tutifornari@gmail.com), [schaub@nics.unicamp.br](mailto:schaub@nics.unicamp.br)

Free improvisation lets the performer to openly explore musical outcomes unbounded by any musical structure or notation. However, the human mind is naturally constrained by its own built-in habits. As such, musicians usually develop, during years of practice and aesthetic predilections, a repertoire of self-known musical patterns which are intentionally, and sometimes unconsciously, retrieved and used by them during improvisations.

This work presents an algorithm that aims to retrieve in real-time similarities during sessions of free improvisation of certain specific sound features so the musician can better manage his repertoire of musical patterns instead of be unconsciously guided by habits.

There are several audio features in music that composers and performers manipulate in order to express its aesthetic intention. Some of the most basic ones (also known as lower-level or non-contextual features) are: Loudness (perception of sound intensity), Pitch (clarity and perception of sound fundamental) and Timbre (perception of the sound dynamic frequency spectral shape and distribution). Lower-level features are free of cognitive ground, also known as Context, thus occurring in a time frame smaller than the notion of “now” in music, also referred as Specious Present [1]. This introductory work focusses on the expectation of a single and simple lower-level musical feature: Loudness.

Equal-loudness contours, also known as Fletcher-Munson curves, empirically describe the relationship between intensity and frequency for a simple (sinusoidal) sound. Loudness is thus dependent on the intensity and frequency of each partial that creates sound. In music production, Loudness is loosely

associated with RMS (Root Mean Square) of the intensity of an audio signal, despite the fact that RMS doesn't take into consideration the frequency of the fundamental (also known as F0) as well as the frequencies of the higher ones that together compound the sound we hear.

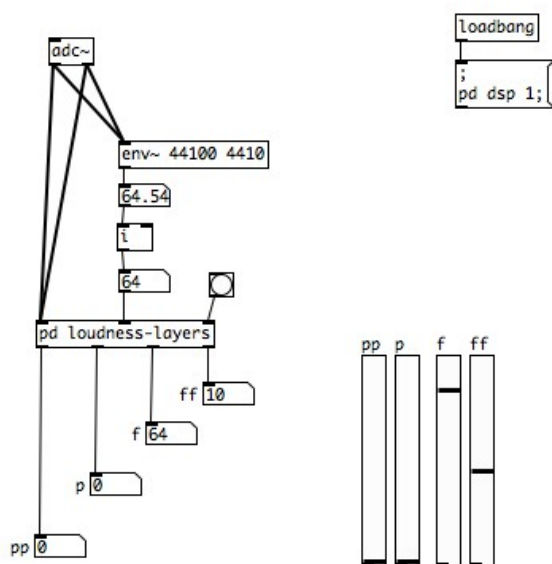
In this work we considered 4 Loudness Levels (LL), which are labeled by their musical terms, in Italian. From quiet to loud, they are: pp (*pianissimo*), p (*piano*), f (*forte*) and ff (*fortissimo*). As the musician performer, in our case, of a melodic musical instrument, carries out his/her improvisation, each note played will presents its own particular LL, that can be quantized into one of the 4 categories mentioned above. As the performance develops along time, the accumulation of LLs for each note played during improvisation is shown in a histogram, thus offering to the performer a graphical dynamic feedback depicting the frequency of each LL occurrence.

For each one of the 4 LLs, a simple Forgetting Curve was also implemented [2]. This function describes the decline over time of human memory as well as cognitive attention for a particular repeating event, such as the LL of sequential notes in an improvised melody. This algorithm, compound of LL histograms increased by similar occurrences of Loudness and decreased by the time lag between similar events (modulated by a simple forgetting curve), aims to introduce a simple model of melodic expectation based on LLs.

Music Expectation is a term used in the field of Music Cognition that refers to the study of listener's automatic (and sometimes unconscious) mental strategies used to predict musical events. The engagement in this

continuous task of listening prediction is referred by David Huron as Anticipation [3]. There are several models of musical expectation described in the literature, such as: the Implication-Realization (I-R) [4], the system and contrast model [5], the Margulis's model of melodic expectation [6]. Huron's general theory of expectation introduces an event timeline known as ITPRA (Imagination, Tension, Prediction, Reaction and Appraisal) which offers an explanation for the fact that music is so efficient to engage the brain into formulating predictions (expectations) of future musical events while listening to music, thus evoking emotions in the listener through mental responses of reward or punishment respectively associated with correct or incorrect musical outcome predictions [3].

This simple algorithm was programmed as a patch in Pure Data (Pd), version 0.47. Pd is a well-known free software platform for the programming of real-time data processing (specially for live performances and multimodal artworks). The figure below shows the referred patch frontend with the implementation of the LL prediction model, within the subpatch (the box in the left side) entitled "loudness-layers".



**Figure 1: The loudness expectation patch**

The four vertical slides, shown in the bottom right side of this figure, correspond to the LL histograms that show in real-time the increasing accumulation and decreasing forgetting factor of each LL during a musical

performance.

This model intends to help the performer to guide the improvisation in order to avoid the excessive repetition of a particular sound feature state (in this case, Loudness), as well as to enable this performer to intentionally repeat a state in order to build up a particular intended emotional state generated by the listener's expectation. In future models other sound aspects can be researched and used to guide the performer during sessions of free improvisation, such as: pitch accumulation, timbre resemblance, pulse pattern detection, and so forth. With such set of tools the performer shall be able to better explore his/her musical endeavors during sessions of free improvisation in order to better manipulate and guide the cognitive process of expectation and anticipation present in the minds of all listeners.

## References

- [1] Andersen, Holly and Grush, Rick (2009) A Brief History of Time Consciousness: Historical Precursors to James and Husserl. *Journal of the History of Philosophy*, 47 (2). pp. 277-307.
- [2] Loftus, Geoffrey R. (1985). Evaluating forgetting curves. *Journal of Experimental Psychology: Learning, Memory, and Cognition*. 11 (2): 397–406. doi:10.1037/0278-7393.11.2.397.
- [3] Huron, David (2006). *Sweet anticipation: music and the psychology of expectation*. M.I.T. Press. ISBN 9780262582780.
- [4] Narmour, Eugene (1992). *The Analysis and Cognition of Melodic Complexity: the Implication-Realization model*. University of Chicago Press. ISBN 0-226-56842-3
- [5] Bimbot, Frédéric; Emmanuel Deruty; Gabriel Sargent; Emmanuel Vincent (June 2012). Complementary report to the Article "Semiotic structure labeling of music pieces: concepts, methods and annotation conventions (Proceedings ISMIR 2012).
- [6] Elizabeth H. Margulis (2005). "A model of melodic expectation". *Music Perception*.

# An open dataset for vocal music transcription

Marcos Voitowitz<sup>1\*</sup>, Helena de S. Nunes<sup>1</sup>, Rodrigo Schramm<sup>1</sup>

<sup>1</sup>Music Department, Universidade Federal do Rio Grande do Sul  
Rua Senhor dos Passos, 248 – Porto Alegre, RS

marcosvoitowitz@gmail.com, helena.souza.nunes@ufrgs.br, rschramm@ufrgs.br

## Abstract

This work presents an audio dataset which is designed to support the development of techniques for multi-pitch detection and voice assignment applied to audio recordings containing performances with multiple singers. The proposed dataset contains recordings of popular Brazilian songs, performed by non-professional vocal quartets. Besides the mix down with the complete ensemble, the dataset also contains each vocal part recorded in separated tracks, with its frame-based pitch ground truth and music score.

## 1. Introduction

The technology of automatic music transcription has evolved significantly; however, a process capable of converting the audio into symbolic representation of a music score is still considered a major challenge, particularly in the transcription of recordings with multiple singers [1]. A growing number of techniques for the transcription of polyphonic signals have been proposed over the last decade [2], especially some more recent, involving machine learning, such as Deep Learning [3]. One of the major challenges for the development of these techniques is the lack of databases for training, testing and algorithm evaluations. This work describes the process of creating a new database of this type, with audio recordings of SATB vocal samples.

## 2. Materials and Methods

The database described here is composed of 114 audio files, captured at the Center for Electronic Music (CME) of UFRGS, between

March and April 2017. For the dataset recordings, we have used ten small excerpts (between four and eight measures) of Brazilian choral works with four voices. These pieces include various musical components and have different characteristics (triplets, punctuated rhythms, tempo changes, alternating metric bars, distinct melodic extension, etc.). The set of choral pieces is: Cabocla Bonita (P. A. Amorim), Canário Terra (F. Matos, 2008), Dies Sanctificatus (J. Maurício, 1793), Divertimento Coral (E. Aguiar, 1950), Final (Guerra-Peixe, 1973), Lá Vai Eu (Guerra-Peixe, 1973), Minha Namorada (C. Lyra, V. de Moraes, D. Cozzela, 2009), Muiraquitã (P. Amorim, A. Diniz, T. Sias, 2015), Padre Nosso (G. Velasquez, 1908) e Síte Pescadores (D. Caymmi, 1957).

The group of interpreters was formed by seven undergraduate students from the third semester of the UFRGS Music Course: a soprano, an alto, three tenors and two basses. All had some previous experience in singing (choir participation and / or accompanied singing groups), but only the soprano was a professional soloist. No person had previous experience in studio work.



Fig 01 – Recording Session

The audio setup used to capture the voices was: a Shure SM58Beta microphone, a Shure SM57

\*Supported by CNPq – Proc. 145275/2016-7



microphone, an M-AUDIO FAST TRACK ULTRA preamp (interface) and the Cockos Reaper v5.27 multi track system software. All recordings were sampled at a rate of 44100Hz / 24bit, and the audio was recorded in WAV format. On each recording session, in addition to the metronome, the performers could listen to a previously recorded piano base, corresponding to the melodic line of their own vocal part. The goal was to ensure that during the recordings each voice had a reference melody to aid in the tuning.

Five recording sessions were held, each lasting five hours. In the first session, a period of fifteen to twenty minutes was provided, so that the students could know the score, in a process of sight-reading. The following sessions were held at intervals of one week so that everyone would study the scores in advance. The quality of each of these two outcomes was very different. The recording process was performed in two stages: 1) sample collection using piano accompaniment as the melodic reference; and 2) sample collection without melodic reference. Both stages have used a stereo metronome inserted into the headset. Each voice was recorded individually, using two takes in sequence: after the recording with metronome and reference melody (first take), the singer restarted singing the same passage (second take), with the metronome, but without the melodic reference.

The first results showed many mistakes and high time consumption, requiring on average three sessions (lasting around fifteen minutes) for each person. Throughout the sessions, due to the study of the scores and to the own experience acquired, a gradual reduction of this effort was observed to the finalization of the audio files, passing for an average of two takes and seven minutes per excerpt / singer. We have identified the users have more difficulties in the maintenance of time than in the precision of the tuning. The recordings have no cut or assembly of takes; On account of a more compromising error, the session was cancelled, and the singer repeated the recording from the beginning.

In addition to the original sound material, the dataset also contains the music score for each piece, and the respective pitch tracking and

automatic melody transcription for each of the vocal parts. This annotation was made automatically using the pYIN [4] algorithm (using default parameters). The resulting files are organized, for each of the pieces and their respective voices, in the following structure:

- **song**
  - song\_mix.wav
- **soprano**
  - song\_soprano.wav
  - song\_soprano\_pitch\_track.txt
  - song\_soprano\_notes.txt
  - song\_soprano\_notes.mid

The final version of this dataset is available at <http://inf.ufrgs.br/~rschramm/projects/msingers/>.

### 3. Conclusion

The search for similar material to the one produced here, already ready to use, proved disappointing. Although individual recordings of choral piece voices have been found, they have the greater purpose of serving as support for the playing of choir's pieces, not serving as data for machine learning techniques. The construction of a database for supporting experiments of automatic music transcription and voice assignment, a direction in which this work will continue, is a necessary and urgent task.

### 4. References

- [1] R. Schramm and E. Benetos. Automatic transcription of a cappella recordings from multiple singers. In AES International Conference on Semantic Audio, June 2017.
- [2] E. Benetos, S. Dixon, D. Giannoulis, H. Kirchhoff, and A. Klapuri. Automatic music transcription: challenges and future directions. *J. Intell. Inf. Syst.*, 41(3):407–434, 2013.
- [3] 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*, 24(5):927–939, May 2016.
- [4] M. Mauch and S. Dixon, “pYIN: A Fundamental Frequency Estimator Using Probabilistic Threshold Distributions”, in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2014.

# Complex networks of chord transitions in Alexander Scriabin's piano pieces

Augusto Paladino, Bruno Mesz, Juan Pégola

MUNTREF Arte y Ciencia – Universidad Nacional de Tres de Febrero  
Valentín Gómez 4752, Caseros, Buenos Aires, Argentina (B1678ABH)

## Abstract

We consider chord transition networks built from piano pieces by Aleksandr Skriabin. We design a random walk algorithm for composing music based in the networks, and present two pieces generated in this fashion.

## 1. Introduction

Recently, complex networks methods have been employed to analyze pitch and timbre transitions, both in individual works [1] and large collections of pieces [2]

We employed this method for analyzing selected piano pieces by A. Skriabin (see Table 1)

From the MIDI files of the pieces, transposed to C (major or minor), we built networks of pitch class (chroma) chords, see Figure 1 for a schematic description.

a)



b)



**Figure 1. Network creation. a) Original score (Mazurka opus 25 n 3, bar 11), b) network nodes and links after transposition to C minor.**

Note that we are only considering chord types and not harmonic functions, so we don't take into account enharmonic spelling.

## 2. Network structure

We fitted the frequencies of chords, sorted in decreasing order (that is, ordered by rank  $r$ , where  $r = 1$  for the most frequent chord and so forth), with a Zipf law of the form  $z = Cr^{-\alpha}$ . For our fitting procedure, we used the approach of Clauset et al [3]. We found nice fits with truncated power laws (see Figure 2). The scaling exponents range from 2.07 to 3.43, comparable to those found in [4] for the distribution of notes in classical music. Usual network measures and metrics are given in Table 1. They stand in strong contrast to other harmonic networks such as those of [2], being far for the small-worldness that characterizes mainstream popular music. Clustering and average shortest path length are similar to those of random networks with the same number of nodes and links.

## 3. Artificial music generation

A simple random walk algorithm in the networks generates a sequence of chords with transition probabilities proportional to the weights (defined as the number of transitions between two given chords in a piece). We generated two piano pieces using this method.

Opus	P-law freq	P-law degree	Sp	r	C	L
Sonate 7	2.78	3.5	0.005	-0.03	0.06 (0.007)	33.17 (13.8)
Op. 25 n 3	3.43	3.16	0.005	0.1	0.03 (0.02)	9.90 (8.45)
Op. 42 n 5	2.62	2.8	0.004	0.1	0.1 (0.03)	5.70 (5.48)
Op. 69 n 2	3.46	3.5	0.01	-0.04	0.03 (0.03)	8.06 (6.46)
Op. 72	2.16	3.5	0.001	-0.008	0.0210 (0.005)	54.45 (14.81)
Op. 45 n 1	2.17	2.68	0.006	0.0008	0 (0.002)	33.06 (20.22)
Op 74 n 3	2.15	3.5	0.01	-0.21	0.02 (0.01)	19.49 (13.16)
Op 25 n 9	2.07	2.51	0.006	0.04	0.04 (0.04)	5.89 (6.16)

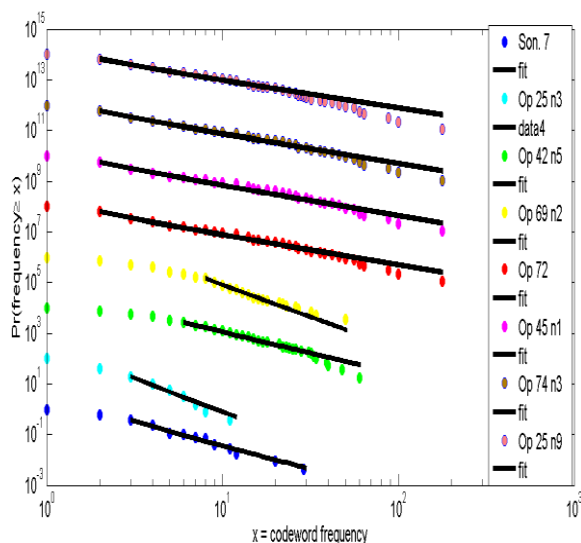
**Table 1. Network measures.** P-law frequency and P-law degree are the fitted exponents,  $S_p$  denotes sparseness,  $r$  the assortativity,  $C$  the average local clustering coefficient,  $L$  the average shortest path length. Coefficients for random networks are in brackets.

### 3.1 First piece<sup>1</sup>

The following composition was created from the above mentioned MIDI scores of the Russian composer Aleksandr Skriabin (1872 – 1915). Rhythm was generated by a random walk in a Barabasi-Albert network of ten rhythmic cells distributed independently in both hands.

### 3.2 Another piece<sup>2</sup>

This composition is based in Op. 69 n 2. Chords are diluted in resonance and fuzzily presented with extremely soft dynamics and complex rhythms.



**Figure 2: Complementary cumulative distribution of codeword frequencies and their fits by power laws. Curves are shifted by a factor of 100 in the vertical axis for ease of visualization.**

## 4. Conclusion

Complex networks give a global picture of first order harmonic transitions, providing information relative to **chord frequency power laws** (the greater the  $\alpha$ , the more limited the vocabulary, with few frequent chords and many

rare ones), **sparseness** (going from 0 to 1; small values mean fewer possible chord transitions, potentially favoring the learning of harmonic expectancies), **chord degrees** (variety of chords following a given one), **clustering** (more clustering means more different ways of reaching neighboring chords from a given one), **shortest average path length** (when this is small, arbitrary pairs of chords are connected by few transitions, in average). These network coefficients appear useful for stylistic comparison between composers and genres [5], and can be applied also to other musical parameters such as timbre. We also began to explore the use of network structures for music generation.

## References

- [1] Gomez, F., Lorimer, T. and Stoop, R. (2014, July). Complex Networks of Harmonic Structure in Classical Music. In International Conference on Nonlinear Dynamics of Electronic Systems (pp. 262-269). Springer International Publishing.
- [2] Serra, J., Corral, A., Bogaña, M., Haro, M., and Arcos, J. L. (2012). Measuring the evolution of contemporary western popular music.
- [3] Clauset, A., Shalizi, C. R. and Newman, M. E. (2009). Power-law distributions in empirical data. *SIAM review*, 51(4), 661-703.
- [4] Zanette, D. H. (2006). Zipf's law and the creation of musical context. *Musicae Scientiae*, 10(1), 3-18.
- [5] Mesz, B., Paladino, A., Pérgola, J., Amster, P. (forthcoming). The evolution of Tango harmony, 1910-1960. To appear in *Mathematics and Computation in Music 2017, Lecture Notes in Computer Science*, Springer.

<sup>1</sup> Available at: <https://drive.google.com/file/d/0B-wA72ZAJgO5eHc4MEIvejZINUE/view>

<sup>2</sup> <https://drive.google.com/open?id=0B-wA72ZAJgO5bnhsN090ZDIPYVU>

# Developing a set of applications for music creation using low-cost brain-computer interfaces

Guilherme Feulo do Espirito Santo , Marcelo Queiroz<sup>1</sup>

<sup>1</sup>Computer Music Research Group – University of São Paulo  
Rua do Matão, 1010 – 05508-090 Sao Paulo, SP

feulo@ime.usp.br, mqz@ime.usp.br

## Abstract

In this paper we present a set of applications to use low-cost brain-computer interfaces for music creation and performance in the form of a PureData (Pd) External Library. This set of Pd objects allows artists and other potentially interested users to access both raw EEG signals and derived features and to include them in their own implementations.

## 1. Introduction

Brain-Computer Interfaces emerged in the medical field as a way to help patients with severe motor disabilities to communicate with their surroundings using only their brain activity [1]. Nowadays, focused in the general public, companies such as Neurosky and Emotiv started to develop more affordable interfaces for applications in entertainment and the arts.

Still in the BCI popularization context, Do-It-Yourself interfaces from hacked toys like MindFlex game appear as accessible alternatives to EPOC or Mindwave interfaces. The website Instructables<sup>1</sup> has lots of interesting tutorials, including one showing how to change the MindFlex operation mode and send its data via Bluetooth to a computer or smartphone [2].

## 2. The Application Set

To make it easier to use Brain-Computer Interfaces for interactive music and multimedia creation, we have developed a PureData<sup>2</sup> (Pd) External Library to work with the hacked MindFlex interface.

In the following, externals are divided in three groups: The main object, Feature Extraction and Auxiliary Functions. The source code and documentation for all the externals can be downloaded here<sup>3</sup>

### 2.1. The Main Object

The `mindflex` object is responsible for establishing the Bluetooth connection between interface and computer and for parsing incoming packets. To connect to an interface the user can either use the search function present in the `mindflex` object, and select the interface from a list of all Bluetooth devices available, or connect directly to the interface by informing the corresponding Bluetooth address.

After start receiving data, the `mindflex` object sends through different outlets all the information provided by the interface: the 12-bit signed integer raw signal sampled at 512 Hz and several signal features computed once every second: attention and meditation percentage levels, and energy levels for the 8 principal EEG bands (each as a 24-bit unsigned integer).

### 2.2. Feature Extraction

The feature extraction step is very important within the Brain-Computer Interface workflow, since it is based on the extracted features that we can infer the user's mental state and perform some associated action. Spectral-content-related features were chosen due to the MindFlex interface characteristic of having only one electrode located on the forehead.

The `mindfft` object calculates the Fast Fourier Transform (FFT) of the EEG signal for

<sup>1</sup><http://www.instructables.com>

<sup>2</sup><https://puredata.info/>

<sup>3</sup>[https://github.com/Feulo/pd\\_mindflex](https://github.com/Feulo/pd_mindflex)

a given window size (a power-of-2 passed as a creation argument). The FFT is calculated by the Cooley-Tukey Radix-2 algorithm, and the outputs are the spectrum real and imaginary components, respectively.

One of the most used features in BCI applications is the energy (sum of squared amplitude values) of specific frequency bands, which can be related to different mind states [3]. The energy in these bands can be calculated using the `mind.filter` object, which outputs the portion of the signal corresponding to a chosen frequency band, in addition to the `mind.energy` object, which receives a window size as a creation argument and produces a new energy value for each corresponding time window of the incoming signal.

### 2.3. Auxiliary Functions

A small set of auxiliary functions were developed to help users during patch creation; these functions do not perform BCI-related tasks but can be useful to test and collect data from patches.

When writing a PureData patch it is often necessary to display the data that is being acquired and manipulated. The `mind.tabwrite` object works similarly to Pd's `tabwrite` and `tabwrite~`, writing incoming data in an array passed as argument.

The `mind.sin` object can be used to simulate the raw data outputted by the `mindflex` object, but instead of outputting a complex EEG signal, it generates a sinusoidal wave with the same sample-rate of the MindFlex interface. The desired frequency is passed as argument and can be changed by sending a message with the new desired frequency. The `mind.sin` can be used to test patch response to a specific EEG frequency band without interference from other signal components.

The `mind.rec` and `mind.play` functions are used, respectively, to record and play a session; the first creates a .RAW file and records all samples outputted by the interface between the start and stop recording messages passed to it, and the `mind.play` loads a recorded session file and outputs it as a compatible EEG signal. The use

of these two functions helps testing patches under development using the same fixed session as input.

### 3. Final Considerations and Future Work

In this work we have presented a PureData External library developed to allow the use of a Do-it-Yourself Brain-Computer Interface for music and multimedia creation. The proposed functions grant access to all EEG data outputted by the interface and provide tools for manipulating this data, allowing new possibilities for composers and performers.

As future work, we aim to explore the usability of this type of interface in different musical and multimedia creation scenarios through existing partnerships with artists using Brain-Computer Interfaces in their works. We also want to investigate the voluntary control level provided by the interface and the extracted features through experiments with volunteers trying to accomplish proposed tasks by actively manipulating their mental states.

### Acknowledgments

The authors would like to thank the NuSom - Sonology Research Center at the University of São Paulo, and the second author acknowledges the funding received from CNPq.

### References

- [1] Jorge Bazarrica Oschoa. EEG signal classification for brain computer interface applications. Master's thesis, Ecole Polytechnique Federale De Lausanne, 2002.
- [2] Instructables. Mindflex EEG with raw data over bluetooth. <http://www.instructables.com/id/Mindflex-EEG-with-raw-data/-over-Bluetooth/>, 2016. Último acesso em 27/05/2016.
- [3] Z Vamvakousis and R Ramirez. Towards a low cost mu-rhythm based BCI. In *Proceedings of the Fifth International Brain-Computer Interface Meeting*, June 2013.

# Live Coding Console with Remote Audience into the web

Guilherme Martins Lunhani<sup>1</sup>, Flávio Luiz Schiavoni<sup>2</sup>

<sup>1</sup>Rua Abolição 403 – 18044-070, Sorocaba, SP

<sup>2</sup>Federal University of São João Del Rei (UFSJ)

São João Del Rei, Minas Gerais, Brazil

lunhanig@gmail.com, fls@ufsj.edu.br

## Abstract

Live coding is a (not so) novel form of performance based on computer programming languages. Since the emergence of Web Audio, several initiatives managed to create applications and programming environments on the web for music programming and, why not, live coding with music in web browsers. Even though running on the web, these environments run an individual live coding application without audience capability. This article describes the context, reflections and the development of a live coding web environment and three approaches to create a live coding environment on the web with audience including an initial implementation and some code development. These approaches are conceptual and can be applied on other tools and expand the live coding capability on the web.

## 1. Introduction

In the context of musical productions made by artist-programmers using the textual programming languages [1], Giovanni Mori [2, p.197] defined the British term *livecoding* as a polyvalent technique of improvisation.

In academic context, the term appeared in 2003 [3], and in 2004, a set of rules was formalized by British artist-programmers[4]. This set of rules do not restrict any language expression. At that time, the artistic artifacts fluctuated between music, the audiovisual and dance.

Many *livecoding* into the web start to emerging from a largely discussion since the W3C added the capability of real time audio processing to web browsers [5]. There are many web applications that can generate, process and analyze audio directly using JavaScript, hiding the basic process into classes and functions.

Under this context, we developed a live DSP console as a single *client-side application*, or in

other words, a prototype of a web CLI (Command Line Interface). This work aims to share the current development thoughts and works focusing the creation of a web live coding environment with remote audience.

## 2. Web live coding environment with remote audience

A basic webaudio application is developed in Javascript and works only in the browser. This approach was used in in most applications existents. A step forward to live coding on the web is to try to create a programming environment where the performance could be listened by an audience like a streaming.

### 1st approach: Audio streams

Our first approach to a Live coding web environment with audience is to keep a connection with the server and create an audio stream from the application output. This approach could use a stream server, like Icecast or ShoutCast to distribute the live performance audio stream. The advantage of using one of these servers is that the audience could use any application that connects to these stream servers and does not have to use a web browser to listen the live coding performance. On the live coding environment, a small change is necessary adding a stream object on the audio route to make an audio stream to the server.

### 2nd approach: Audio Stream + code sharing

Our second approach is to send the code developed on the environment with the audio parameters and configuration to the server. The server would run the code locally and send the audio output to audio stream instead of sending it to audio output.

Thus, clients could connect to server audio stream, that also can be an Icecast / Shoutcast

instance, and hear the living performance. The main difference here is a reduced bandwidth between the live Coder and the server.

### 3rd approach: Code Sharing

Our last approach is to send the code to the server to be shared with the audience. In this approach, the audience computer should run the same application as the live coder and audience computer would synthesize locally the audio from the shared code. This approach uses less bandwidth and allow the audience to interact with the source code and learn how the code was written. The server implementation is close to a white board, a server to share text among different clients. Since the server does not need to run Javascript code, it can be implemented in any serve-side language like PHP or JSP.

### 3. Approaches comparison

The first comparison among the three approaches is the shared data type. On the first approach, it is shared audio, the second approach shares code between the live coder and the server and audio between the server and audience and the third approach shares only code.

Since the shared data type is different, on the first approach the code is private and the live coder does not need to share it. The second approach is semi private because it is sent to the server and the server could save the code, publish it to the audience during the performance or later. The audience could read the code but a change on the code would not change the local sound since it is not processed locally but received already processed. On the third approach the code is shared and real time published to the audience. The audience could change the code and affect the hearing experience and also re share the code on the web.

The presence of the code can change the hearing experience. Both, first and second approach leads to a passive hearing while the third approach allows the code reading with the listening. Audience can notice programming changes, code errors and typos and it can affect the hearing experience.

Lastly, the client program influences the hearing experience but can simplify the audience lis-

tening. First and second approaches can use a stream cast application, like VLC, while the third approach is more restrictive and demands a browser as the client application.

### 4. Conclusion

This paper presented 3 approaches to create a live coding web environment with audience. These approaches are being developed our work in progress, referring to the works of Pietro Grossi and Max Mathews, considered by these authors as premature live coding cases.

We planned to stream or pipe the audio to a streaming server, by a streaming server channel choose by the user, or by WebRTC, but this feature is incomplete. The presented application is under development<sup>1</sup> and has, among the complete and incomplete features, a server that saves a code and processes it into an audio file. Our current implementation reflects an agreement between the second and third approaches.

It is part of Future works to implement the three approaches here presented to have a better framework to test and validate user's experience in each scenario.

### References

- [1] Alex McLean. *Artist-Programmers and Programming Languages for the Arts*. PhD thesis, Department of Computing, Goldsmiths, University of London, October 2011.
- [2] Giovanni Mori. Analysing live coding with ethnographical approach. In *ICLC2015 Proceedings*, pages 117–124, 2015.
- [3] Nick Collins, Alex McLean, Julian Rohrhuber, and Adrian Ward. Live coding in laptop performance. *Organised Sound*, 8(3):321–330, 2003.
- [4] Adrian Ward, Julian Rohrhuber, Fredrik Olofson, Alex Mclean, Dave Griffiths, Nick Collins, and Amy Alexander. Live algorithm programming and temporary organization for its promotion, 2004.
- [5] Lonce Wyse and Srikumar Subramanian. The viability of the web browser as a computer music platform. *Computer Music Journal*, 37(4):10–23, 2013.

<sup>1</sup>Available on:  
<https://github.com/lunhg/termpot>.

# Melody and accompaniment separation using enhanced binary masks

Shayenne da Luz Moura<sup>1\*</sup>, Marcelo Queiroz<sup>1</sup>

<sup>1</sup>Computer Science Department, University of São Paulo

shayenne@ime.usp.br, mqz@ime.usp.br

## Abstract

Recovering melodic information from sound signals has several applications, being an important task in Computational Auditory Scene Analysis. This work presents enhancement methods for filtering the spectrogram of an audio signal based on melodic annotations for the separation of melody and accompaniment in different audio tracks. Preliminary quantitative results correlate well with subjective evaluations, showing that enhanced binary masks provide a reasonable starting point for the refinement of automatic melodic separation strategies based on spectrogram resynthesis.

## Introduction

This work deals with the processing of musical signals aiming at the extraction of melodies based on annotated melodic transcriptions. The main contribution is the proposition of enhanced harmonic binary masks aimed at preserving important timbral and transient characteristics of the melodic instrument being extracted.

The input audio signal  $x$  is first transformed into a time-frequency representation in the form of a spectrogram  $\mathcal{X}(m, k)$  where  $m$  and  $k$  are frame and frequency indices. This spectrogram is invertible by taking the IFFT of each audio frame and overlap-adding the results. The goal is to define a binary mask  $\mathcal{B}(m, k)$  that acts as a spectrogram filter allowing the decomposition of the signal in two parts,  $\mathcal{X}_{\mathcal{M}} = \mathcal{B}\mathcal{X}$  and  $\mathcal{X}_{\mathcal{A}} = (1 - \mathcal{B})\mathcal{X}$ , in such a way that the resynthesis of  $\mathcal{X}_{\mathcal{M}}$  contains the melody and the resynthesis of  $\mathcal{X}_{\mathcal{A}}$  contains the accompaniment.

By knowing the spectrogram and an annotated melodic line, given by either a human expert or as the result of an automatic pitch tracker such as Melodia [1], it is possible to create a crude binary

mask containing only the spectrogram bins corresponding to this F0-profile and use it for resynthesis. Of course most melodic instruments will produce harmonic spectra, so extending this binary mask to include all the harmonic components of this F0-profile is a natural idea. Once a binary melodic mask is defined, the remaining spectrogram bins would produce the accompaniment through resynthesis, using the binary complement of the melodic mask.

It is known that distinct sound sources have different spectral behaviors: some sources have smooth onsets while others produce many transient noises at the attack section which are important for the characterization of their timbre; These individual timbral traits of music sound sources demand the refinement of melodic masks so that the audio signal obtained through resynthesis preserves these characteristics.

## Melodic mask enhancements

In order to achieve perceptually expressive results, a number of binary masks were proposed and compared starting with the annotated F0-profile provided in the input: (1) Melodic mask selecting spectrogram bins of the pure F0-profile; (2) Harmonic Mask containing the F0-profile and its higher harmonics; (3) Dilated harmonic mask including upper and lower neighboring bins; (4) Adaptive dilated harmonic mask with spectral width given by a spectral novelty function [2]; (5) Adaptive percussive dilated mask, as the previous mask but expanding only through percussive<sup>1</sup> bins (see example in Figure 1); (6) Adaptive dilated mask with backward spreaded onsets<sup>2</sup>; (7) Adaptive percussive dilated mask with backward spreaded onsets.

<sup>1</sup>Based on percussive + harmonic binary spectrogram masks [2].

<sup>2</sup>Wherever an onset is detected in a harmonic partial this mask spreads and dilates through bins of previous frames to collect transient/noisy elements

\*Both authors acknowledge their support by CNPq.



After resynthesis from the filtered spectrogram it is possible to study the perceptual impact of each one of the strategies for melody extraction. Whenever a ground truth is available (e.g. in multichannel recordings where the audio of the melodic instrument is known) it is also possible to correlate perceptual data with objective measurements.

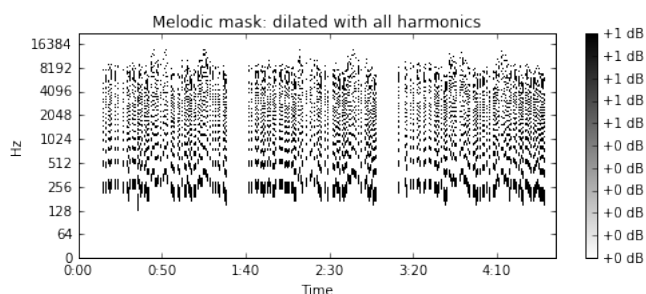


Figure 1: Example of mask of type (5)

## Implementation and evaluation

The melody-accompaniment separation system was developed in Python, using specialized libraries such as *libROSA*<sup>3</sup> for audio processing and *SciPy*<sup>4</sup> for image processing. The Jupyter Notebook environment was used to present the results in an interactive way.

The MedleyDB dataset [3] was used to evaluate the system. Five audio signals were selected which had different musical styles and contained annotated F0-profiles and also separate audio tracks for individual instruments. These were used to evaluate the melody-accompaniment separation, both objectively and perceptually, using all the proposed enhanced masks.

For each piece, the spectrogram of the isolated melodic instrument is used as ground truth. The correlation between each mask and ground truth is calculated, along with *accuracy* and *signal level*. These are defined similarly to the well-known information retrieval measures precision and recall, but refer to energy values: accuracy is defined as how much of the spectral energy of the melodic instrument was retrieved from the melodic mask, and signal level is defined as the part of the recovered energy that was actually

<sup>3</sup><http://librosa.github.io>

<sup>4</sup><http://docs/scipy.org>

part of the melodic instrument. These values are summarized via their harmonic mean (akin to the F-measure in information retrieval, see Figure 2).

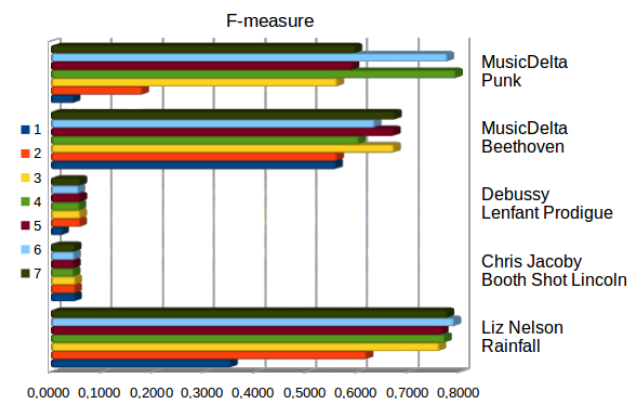


Figure 2: Results for enhanced masks

The subjective evaluation consisted of the perceptual evaluation of the extracted melodies. It was verified that subjectively preferred melodies were generated by masks with higher accuracy or harmonic mean values.

## Conclusion

This work investigated melody-accompaniment separation based on enhanced binary masks. Comparison of objective and subjective results showed that these masks are able to filter out the melodic instrument from polyphonic musical contexts, but each enhancement might be better suited to different inputs according to timbral characteristics of the corresponding melodic instrument.

## References

- [1] J. Salamon and E. Gómez. Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(6):1759–1770, Aug. 2012.
- [2] M. Müller. *Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications*. Springer International Publishing, 2015.
- [3] R. M. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam, and J. P. Bello. MedleyDB: A multitrack dataset for annotation-intensive MIR research. In *15th Int. Soc. for Music Info. Retrieval Conf.*, pages 155–160, Taipei, Taiwan, Oct. 2014.

# Methods on Composer Identification Using Markov Chains

Adilson Neto<sup>1</sup>, Rodrigo Lisboa Pereira<sup>2,1</sup>

<sup>1</sup>Centro Universitário do Estado do Pará – Belém – Pará – Brasil.  
Grupo de Estudos Temático em Computação Aplicada (GET-COM)  
Laboratório de Inteligência Computacional (LINCOMP)

<sup>2</sup>Universidade Federal do Pará – UFPA – Belém – Pará – Brasil

almeidneto@gmail.com, rod.lisboa@gmail.com

## Abstract

Markov chains along with other algorithms have already been used on the identification of music composers. This paper presents a survey on two different types of music coding schemes and the effects of using each one on the precision of the Markov model.

## 1. Introduction

Artistic fields such as music are widely viewed by the general public as processes intrinsic to humans, where algorithms and machines have little aptitude, but despite public opinion, the field of Computer Music is continually growing in subjects like composer identification and algorithmic composition.

Significant work has been done on composer identification using the idea of Markov Chains[1], this paper does not intend to confirm this idea, but to improve it by changing the way a note is mapped to a state on the Markov Model, using a different coding scheme.

A general view of Markov chains will be presented on section 2, section 3 describes the coding schemes used, section 4 documents the methodology and results are discussed on section 5.

## 2. Markov chains

A Markov chain can be interpreted as a random walk through a set of states, where the probability of going to a state depends exclusively on the actual state, in fact, a Markov chain is mathematically characterized by its state-transition matrix [1].

In the context of music and considering it a random process, the notes would be the states and the transitions would be probabilities of a note happening after another as in[1].

## 3. Music Coding Schemes

Two coding schemes were used on this work, the first one represents a music as a sequence of notes, in this scheme, every note is represented as a number and the mathematical difference between two notes is the distance in semitones between them. Giving the value of 20 to the note C, the C major scale would be represented by: 20, 22, 24, 25, 27, 29, 31, 32.

The second coding scheme here proposed for composer identification also represents a music as a sequence of numbers, but these numbers are not notes, but the tonal distance between each note, so the C major scale would be represented by: 0, 2, 2, 1, 2, 2, 2, 1. Note that the sequence always starts with zero, since there is no previous note to create a tonal distance from.

Both these coding schemes were analyzed on the work of Cruz and Vidal[2], but applied to musical style recognition utilizing grammar induction, where the differential scheme got some of the best results, mainly because scales and patterns are better matched on this notation, since on a differential notation every major scale would be considered the same sequence.

## 4. Methodology

For the experiments of this paper, 156 piano scores were used from 7 different composers:

Albéniz, Haydn, Mozart, Schubert, Schumann, Grieg and Mendelssohn.

The results of this paper were obtained by three different tests, each of them involving four different composers, for each test, a Markov model was created for every author based on half of the music files (the training set) and then, these models were used to try to determinate the original author of the remaining files (the testing set), every time a model predicted its original author correctly, this was considered a positive.

## 5. Results

Figures 1, 2 and 3 display the results of the tests, where the vertical axis indicate the percentage of positive results, where the algorithm correctly predicted the original composer and the horizontal axis represents the corresponding original author.

After analyzing the results, it can be seen that the difference in coding schemes is leading to different results, although the differential scheme is not always giving the best result, it commonly improves over the sequential scheme.

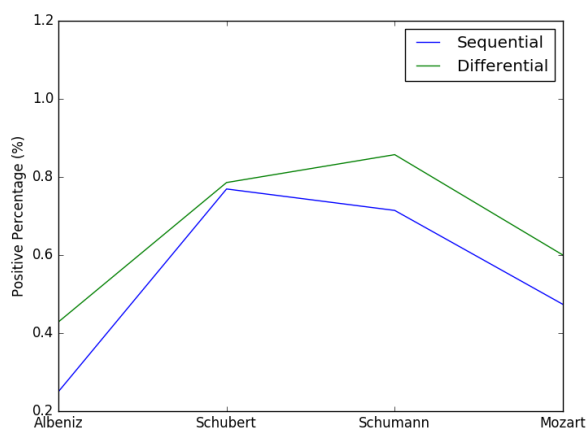


Figure 1: First test

## 6. Conclusion

There remains many possibilities to be explored on the field of composer identification and music recognition, but the notation appears to be a stepping stone on the improvement of any algorithm dealing with music. This is to be expected,

as the only way an algorithm deals with music is through the data we feed it with, and the structure of this data changes the way a algorithm deals with the music. Further work can be developed on the improvement of the coding scheme and its use.

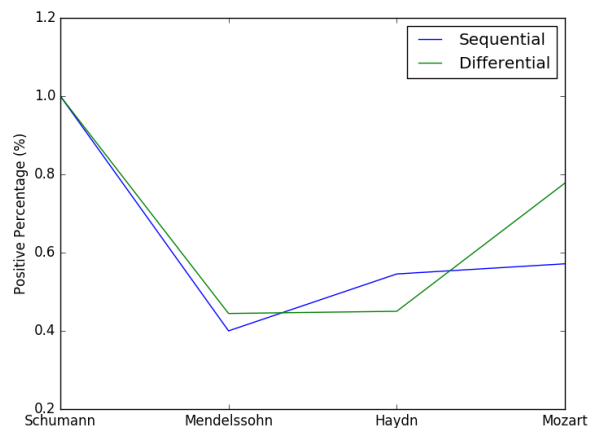


Figure 2: Second test

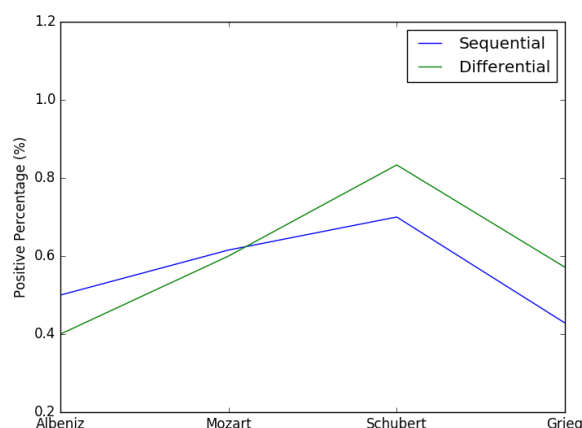


Figure 3: Third test

## References

- [1] Yi-Wen Liu and Eleanor Selfridge-Field. Modeling music as markov chains: Composer identification, 2002.
- [2] Pedro P. Cruz-Alcázar and Enrique Vidal-Ruiz. *Learning regular grammars to model musical style: Comparing different coding schemes*, pages 211–222. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.

# New developments on the augmentation of a classical guitar: Addition of embedded sound synthesis and OSC communication over network

Eduardo A. L. Meneses<sup>1</sup>, Marcelo M. Wanderley<sup>1</sup>

<sup>1</sup>Input Devices and Music Interaction Laboratory (IDMIL)  
Centre for Interdisciplinary Research in Music Media and Technology (CIRMMT)  
McGill University  
550 Sherbrooke St. W. – Montreal, Quebec, Canada

eduardo.meneses@mail.mcgill.ca, marcelo.wanderley@mcgill.ca

The classical nylon string guitar is a versatile musical instrument that can generate a wide variety of timbres, but other characteristics such as the short sustain and the lack of sound intensity control after the attack are usually considered severe restrictions imposed by the physical structure of the instrument.

One of the approaches to solve these problems is the construction of augmented musical instruments (AMIs). Using sensors and actuators it is possible to generate gestural data that can be used to control specific software programmed to address these issues using digital signal processing (DSP).

The *GuitarAMI* aims to use gestural data to control algorithms that overcome the restrictions described above [1]. *GuitarAMI* explores possibilities of modification of the commonly considered restrictive intrinsic characteristics of the acoustic guitar by using sensors that generate data through effective and free gestures [2], controlling sound manipulation patches programmed in Pure Data [3]. The Pure Data patch modifies the sustain time by performing a Fast Fourier Transform (FFT) to analyze the audio signal and later re-synthesizes a sound that can be sustained indefinitely.

One of the major problems encountered during the use of previous *GuitarAMI* prototypes was the difficulty of system setup for performances. In the first prototypes the number of connections and cables increased the possibility of malfunctioning and made the instrument less robust. With the third prototype there was some improvement regarding setup time but the instrument still had not reached the robustness required for plug-and-play use by performers.



**Figure 1: GuitarAMI new prototype, constructed using the ESP8260 WiFi microcontroller, ultrasonic sensor and accelerometer.**

Embedding a single-board computer and an audio interface into the *GuitarAMI* base can simultaneously increase robustness and make the AMI more usable by the performer. We chose the *Raspberry Pi 3 model B* due to its unique qualities: As it is an open-source hardware platform we have a greater range of compatible boards and components. In addition, the *Raspberry Pi*

can run a wide range of operating systems including several Linux flavors, Android and even Windows 10. Finally, the Raspberry Pi has a wide and active user community, providing compatibility with other digital musical instrument (DMI) projects.

The latest GuitarAMI prototype uses a Raspberry Pi running the official Linux distribution entitled *Raspbian Jessie*, released in November 2016 [4]. This single-board computer is responsible for running the GuitarAMI embedded sound synthesis algorithms.

In GuitarAMI's older prototypes an Arduino was used to send gestural data over a serial-USB connection to a computer running a Pure Data patch that contains the processing, sound manipulation and synthesis algorithms. With the Raspberry Pi implementation the patch had to be re-programmed for compatibility with PD Vanilla (version officially maintained by Miller Puckette).

Currently, communication using Open Sound Control (OSC) is fully implemented, allowing GuitarAMI to send already processed gestural data over network to any other device.

Along with the hardware implementation there was also the necessity of updating the GuitarAMI algorithm in order to address the problems reported in [5] such as timing, similarity between the synthesized and real sounds and volume control. In the improved GuitarAMI patch we used techniques commonly applied by the phase vocoder to accurately control the execution speed of a real time sound buffered into two different arrays. This process should be performed with minimal latency possible so that it can be used in real-time performances.

The algorithm design uses the performance model presented by Young and Lexer in [6], where gesture and audio analysis are parameters subordinates to creative decision making, i.e. the processes depends on performer's decisions in real time. The interface design and in a later mapping for use in AMIs must take this model into account.

Pure Data *rfft~* and *irfft~* objects are responsible for these operations and they use the block

and overlapping settings configured for the sub-patch. These settings can be changed to provide the best relationship between frequency and time resolution. As expected, signals are handled as pairs containing real and imaginary part.

With the use of Raspberry Pi it was possible to embed a microprocessor powerful enough to perform synthesis and manipulation sound processes already present in previous GuitarAMI prototypes. We performed experiments with two audio interface possibilities and implemented the standard MIDI protocol, allowing communication between GuitarAMI and any hardware or software capable of sending or receiving MIDI data.

## References

- [1] Eduardo Aparecido Lopes Meneses and José Fornari. *Guitarami: um instrumento musical aumentado que transpõe restrições intrínsecas do violão*. In *Proceedings of the 15th Brazilian Symposium on Computer Music*, Campinas/SP, 2015. Sociedade Brasileira de Computação - SBC.
- [2] Claude Cadoz and Marcelo M. Wanderley. *Gesture-music*. In Marcelo M. Wanderley and Marc Battier, editors, *Trends in Gestural Control of Music*, pages 71–94. Editions IRCAM – Centre Pompidou, Paris, 2000.
- [3] Pure Data. *Official web site (pd-extended and community)*. <http://puredata.info/>, abril 2015. [Online; accessed 20-April-2016].
- [4] Raspberry pi homepage. <https://www.raspberrypi.org/>, 2017. [Online; accessed 10-November-2017].
- [5] Eduardo Aparecido Lopes Meneses. *Guitarami: desenvolvimento, implementação e performance de um instrumento musical aumentado que explora possibilidades de modificação de características intrínsecas do violão*. Master's thesis, Universidade Estadual de Campinas (UNICAMP), 2016.
- [6] Michael Young and Sebastian Lexer. *FFT Analysis as a Creative Tools in Live Performance*. In *6th Int. Conference on Digital Audio Effects (DAFX-03)*, pages 6–9, London, 2003. University of London.

# Pedal Board Approach to Sound Effects Customization

Thiago Felipe de Miranda Arcanjo <sup>1</sup> \*, Franklin Magalhães Ribeiro Junior <sup>2</sup>, Tarcísio da Rocha <sup>1</sup>

<sup>1</sup> UFS - Universidade Federal de Sergipe, Brasil, SE

<sup>2</sup> IFMA – Instituto Federal de Educação, Ciência e Tecnologia do Maranhão, Brasil, MA

tf.arcanjo@gmail.com, franklin.mr3@gmail.com, tarcisiorocha@gmail.com

## Introduction

In music sphere, it is recurrent the exploration of new sonorities through the treatment of sound signals with the use of equipment such as pedals, a device that allows to make several changes in the sound signal.

With the advent of applications such as digital audio signal processing systems, software which simulate pedals and other equipment used by musicians, present an attractive alternative because of the possible economic viability inferred. Based on this, we propose in this work a solution designed to run on a PC running Windows operating system that aims to be used with ease and flexibility in accordance with the necessity of the user, since it allows the user to encode, use third-party sound effects and other VST (Virtual Studio Technology) plugins or load the application as a plug-in in a DAW (Digital Audio Workstation) software. In addition, the proposed solution is open source, free and able to process guitar, bass and other instruments.

## Related Solutions

This section presents a brief review of some software related to the software proposed in this work. Some software reviewed were: GNUitar [4], CP Guitar Effect Processor [2], Amplitube [1], ToneBytes Pedals [8], Rakarrack [7], DK Guitar [3].

The solutions in [1] and [8] simulates studio pieces of equipment such as racks, pedals and amplifiers, both are oriented to process guitar and bass sounds. Although [8] works with the same principle of [1], it does not simulate actual models of studio equipment. The solutions in [2], [3], [4] and [7] simulate most of the audio effects used by

guitar players, unlike [1] and [8], they do not simulate equipments.

The solutions reviewed has a free license for usage, yet, only [2], [3], [4] and [7] are open source solutions.

## Proposed Solution

In this work, we propose a piece of software which simulates a set of pedals composed of: compressor, volume, distortion, chorus, flanger and delay in which, during its execution, the user introduces the audio and the software processes it in real time.

To use the application, the user will need to connect its instrument to the input of an audio interface (connected to the computer via USB). In the output interface, the user can use a headset or an amplifier.

When starting the application, it will be displayed to the user a set of pedals with the sound effects mentioned before, which it can be used individually or chained. Figure 1 shows the application feature screen.

As shown in Figure 1, the application shows by default the group of effects previously cited, however, if the user is interested in some sound effects not available in the default configuration, it is possible to create it using the APIs: Pyo [6] and WxPython [9] or the third-party audio effects plugins.

By pressing the “OFF” button, a pedal is activated. To use more than one pedal (sound effect), the other pedals simply need to be activated and the sound will be chained.

It is still possible to notice in Figure 1 that each effect has its own attributes whose intensity can be controlled by knobs. In the case of the chorus effect, for example, its attributes are feedback, depth and balance.

In addition, there is a sequence of buttons called “Swapping Effects” and below it, there are other two buttons, one’s function is to save some preset configuration and the other to load some preset configuration.

The first sequence of buttons cited allows the user to change the position of the pedals and thus change the sound chaining.



Figure 1: Software proposal for sound processing.

## Final Considerations

In this work, we proposed a software in which, based on the applications presented in *Related Solutions*, simulates some of the most popular sound effects among musicians according with [5] and aims to be flexible by allowing users to encode your own audio effects, use third-party audio effects and load the software as a plugin in a DAW software. In addition, it was decided that the software proposed should have a simple graphical interface, be open source and free of charges.

For future work, a comparison of the proposed solution against the others mentioned in *Related Solutions* will be done to evaluate its usability and thus provide improvements in the user experience. In addition, it is also intended to facilitate the user experience who wishes to encode new sound effects through the dynamic addition of sound effects. Furthermore, it will be possible for the user to add sound effects to the pedal set making minimum changes in the source code of the project.

## References

- [1] Amplitude. Amplitude. Available at: <<http://www.ikmultimedia.com/>>. Accessed on: 04/10/2017.
- [2] CP Guitar Effect Processor. CP Guitar Effect Processor. 2017. Available at:

<<https://www.sourceforge.net/projects/cp-gfx/>>. Accessed on: 07/03/2017.

- [3] Daniel Krueger. Tratamento de Sinais Sonoros no Computador Simulando Pedais Virtuais. Trabalho de Conclusão de Curso (Graduação em Ciência da Computação)-Centro de Ciências Tecnológicas da Terra e do Mar, Universidade do Vale do Itajaí, Itajaí, 2008.

[4] GNUitar. GNUitar. 2017. Available at: <<https://sourceforge.net/projects/guitar/>>. Accessed on: 07/03/2017.

[5] Joshua D. Reiss and Andrew P. McPherson. Audio Effects Theory, Implementation, and Application. CRC Press, 2014.

[6] Pyo. Dedicated Python Module For Signal Processing. 2017. Available at: <<http://ajaxsoundstudio.com/software/pyo/>>. Accessed on: 07/03/2017.

[7] Rakarrack. Rakarrack. Available at: <<http://rakarrack.sourceforge.net>>. Accessed on: 07/03/2017.

[8] ToneBytes Pedals. ToneBytes Pedals. Available at: <<http://tonebytes.com/pedals/>>. Accessed on: 07/03/2017.

[9] WxPython. WxPython. 2017. Available at: <<http://www.wxpython.org>>. Accessed on: 07/03/2017.



# T2M: Sonifying Text Using Audio Tags

Bruno Mesz<sup>1\*</sup>, Lucas Samaruga<sup>2</sup>

<sup>1</sup> MUNTREF Arte y Ciencia – Universidad Nacional de Tres de Febrero  
Valentín Gómez 4752, Caseros, Buenos Aires, Argentina, (B1678ABH)

<sup>2</sup> EUDa, Universidad Nacional de Quilmes – CONICET  
Roque Sáenz Peña 352, Bernal, Buenos Aires, Argentina (B1876BXD)

bruno.mesz@gmail.com, samaruga.lucas@gmail.com

## Abstract

We present T2M, a generative text-based music system. T2M takes words from tweets produced by the audience or Twitter™ trending topics and composes music using Freesound™ sound-files labeled with those words as raw material.

## 1. Introduction

T2M explores intra-sonic and extra-sonic meaning of sounds given by tags of audio files in the Freesound database for generating soundscapes from textual input. The motivation for this installation was our work on music-taste crossmodal correspondences [1], which studied musical representations of the extra-musical semantic domain of taste words.

Previous systems mapping text to music are for example [2] and [3]. Semantics of music in connection with language is a vast topic, see for instance [4]. Language is considered an important factor in crossmodal associations [5].

## 2. T2M SuperCollider Quark

The base implementation for this installation [6] is made with SuperCollider [7] as an external library (quark in SC jargon). It consist in a set of classes that organizes information retrieval, both from Freesound [8] and Twitter [9] using external command-line utilities on Linux, processing and storage of that information, sequencing, sound synthesis and visual feedback of original messages and retrieved sounds.

An autonomous T2M execution loop is schematically as follows: tweets query, text parsing and storage, sound query based on selected words, sound-files pull and storage, sound processing and reproduction and visual presentation of information.

Freesound querying and sound-file download depends on the Freesound.sc quark [10] which is a higher level interface to communicate with the Freesound API from within slang (SC programming language). Twitter CLI [11] program is used for querying tweets using Twitter's API. A basic text parsing consisting in noun and verb selections is performed by a command-line utility provided by Apertium [12]. Because performance attendees were Spanish-speaking and due to that Freesound metadata is predominantly in English the translation platform Aперitum was introduced to locally translate information before querying.

### 2.1 System Description

Being an interactive performance, the latency of the involved systems with respect to the attendees input was an important factor to consider regarding sound and visual feedback as an aesthetic proposal. The latency of the system relies mainly on Twitter server's refresh rate and sound-file download time. Former latency is not possible to control but sound data transfer time was reduced by limiting sound-file length by search, downloading compressed (ogg) files and reusing duplicates.

These traits provide processing

\* Supported by Programa de Investigación Sistemas Temporales y Síntesis Espacial en el Arte Sonoro, EUDa, UNQ.



constraints that affect possible outcomes. To work with these system the idea was to use the metaphor of a diffuse cloud of sounds of things happening within a dilative time span considered as the present (i.e., 30 seconds). A given input from attendees do not generate instant discrete sound events but contributes to a sound landscape which runs as far as there is new input data in which the present carries the near past. Because the installation uses flexible network APIs data search was possible either by defining specific tags, so input come from the present public, or external phenomena such as trending topics. The cloud metaphor applies consistently to both cases.

Sound synthesis is based on temporal granularity and spatial diffusion (originally stereo but it can be easily expanded for different setups). Due to some arbitrariness implicit in the search by (user provided) metadata and the unstructured nature of the Freesound database, retrieved sounds were often as different as complex sequences, e.g., field recordings, instrumental samples and loops or, in the most annoying cases, plane sinusoids with full amplitude. Thus the reproduction of these raw materials was transformed by random selection of different moments panned to different positions with different gain. In this way, the resulting mix of different sound-files creates a continuous diffuse texture which varies slowly with the input data evolution.

### 3. Performance at the MAR

We presented T2M at the MAR museum in Mar del Plata, Argentina. Four specially designed perfumes were successively spread in the auditorium, and the audience was required to tweet any associations evoked by the aromas. The tweets were displayed on a screen while T2M generated a sonic atmosphere.

### 4. Conclusions and Future Work

In this installation, different sets of information are set up in real time to reflect an ongoing human experience of the world.

After generating an olfactory atmosphere in the room, we ask the audience to exteriorize, via Twitter, free verbal associations produced by the time-varying smell. The installation then elicits a “music of concepts”, drawing on symbolic meaning generated by arbitrary user defined tags from Freesound that match the words in the tweets.

Future plans include generating soundscapes for a wine tasting and from Twitter trending topics. Additionally, we will implement the analysis of the tweets using natural language processing methods, to gather information on the role of semantics in crossmodal associations [5].

### References

- [1] Mesz, B., Trevisan, M. A., & Sigman, M. *The taste of music. Perception*, 40(2), 209-219. 2011
- [2] [http://www.erikbunger.com/html/let\\_them\\_sing.html](http://www.erikbunger.com/html/let_them_sing.html)
- [3] <http://melobytes.com/app/melobytes>
- [4] Zbikowski, L. M. *Conceptualizing music: Cognitive structure, theory, and analysis*. Oxford University Press. 2002.
- [5] Spence, C. *Crossmodal correspondences: A tutorial review. Attention, Perception, & Psychophysics*, 73(4), 971-995. 2011.
- [6] <https://github.com/smrg-lm/ttm>
- [7] McCartney, J. *Rethinking the Computer Music Language: SuperCollider*. *Computer Music Journal*, 26(4), 61-68, 2002.
- [8] <http://www.freesound.org>
- [9] <https://twitter.com>
- [10] <https://github.com/g-roma/Freesound.sc>
- [11] <https://github.com/sferik/t>
- [12] [http://wiki.apertium.org/wiki/Main\\_Page](http://wiki.apertium.org/wiki/Main_Page)

**Art**



# Artistic Proposal for SBCM 2017

## “Allure”, a *machinic* performance

André L. Martins\*

<sup>1</sup> *Music Department, ECA, Universidade de São Paulo  
Av. Prof. Lúcio Martins Rodrigues, 443, São Paulo - SP, 05508-020*

andremartins@usp.br

### Abstract

Our goal is an artistic performance from the constitution of a complex and diversified environment, convenient for theoretical and practical research on free improvisation. We seek to add theoretical reflection to a type of improvisation that takes place in a hybrid environment that includes, besides traditional musical instruments - acoustic or electronic - other apparatuses and technological tools. The connection of all these devices dynamically and interactively engaged by performers in a given space-time environment is what we call the "hybrid machine" here. This complex environment, which includes all and any sound as raw material, prioritizes the use of new technological digital resources of a mobile character that contribute to enrich the creation plan and interaction between the musicians. At the SBCM 2017 congress, we propose a demonstration of the "operation" of this complex environment through improvisation performance, acoustic instruments, interfaces, mobile computers, microphones, control pedals and the performance environment itself.

### 1. The creative performer

One of our research goals is to put into practice the idea of a creative performer who moves away from rigid musical systems and idiomatic borders, inserted in the complex scenario of a hybrid machine in which it is one more among the various components. For this reason, it is important to enable the performers, their respective acoustic instruments, mobile computers, interfaces, microphones, software,

control pedals, amplifiers, in an environment of free improvisation.

### 2. “Allure”, a *machinic* performance

Structured from our work as a researcher, composer and performer, in the field of the arts and who work in the line of Sonology in the area of Processes of Sound Creation of PPGMUS-USP, the idea of "hybrid machine" (comprising, as previously mentioned, Performers, acoustic instruments, mobile computers, microphones, interfaces, controller pedals, software and amplifiers) is central to our ongoing PhD project.

The research project is based on the following premise: we start from the perception that free improvisation "confronts" music as a machine that opens itself to new and infinite updates, maintaining the desire not to submit to languages and systems, without, however, ignore that it is impossible to start from a zero degree. Thus, in the present project, these individual hybrid machines are fed by the

\* Supported by CAPES

knowledge base (cf. Pressing, 1987<sup>1</sup>) of the performer, in our interactions with this acoustic / digital environment, being possible to describe the potentialities of each element and set, observing the continuous transformations in which there are unfolding of the sound flux and its molecular variations, often imperceptible at first, but which are constituted during the performances and which are observed later in possible records and from analytical processes.



### 3. “Allure” - program notes

Through the creation of a hybrid machine, where the performer seeks to relate to the interfaces, softwares, applications, controllers and musical instrument both in the production of sound and in its manipulation, in addition to the dual relation with the other sounds that are coming from the acoustic instrument and the processed sound material, *feedbacking* by the sounds and musical gestures covered during the performance.

*Instrumentation:* acoustic/electric guitar, digital interfaces, Macbook and iPad.

*Approximate performance time:* 20 minutes.

### 4. Short biography

**André L. Martins:** Professor, composer and researcher, graduated in guitar from LACM (Los Angeles College of Music), is currently PhD student in Music at USP, having 4 albums of instrumental music released. He is a Master of Arts from USP, and participates in Nusom

(*Núcleo de Pesquisas em Sonologia*). His current doctoral research (2016 - present) deals with the composition and use of hybrid machines in free improvisation, which encompasses acoustic and digital.

[www.andremartins.com.br](http://www.andremartins.com.br)



### 5. Setup time and input list

Setup time: about 30-40 minutes.

Technical requirements:

- 2 active speakers
- 1 mixer, 4 or 8 channels
- 2 outlet points

See the stage plot attached on a separate file.

### 6. Intended venue

Evening concert

<sup>1</sup> BORGIO, David; KAISER, Jeff (2010): *Beyond the Ventres: Musical Avant-Gardes since 1950*, p.1. Available at: <http://btc.web.auth.gr/>

# ***Colour Etude I***

**Omar Peracha**

London, Great Britain

omar.peracha@gmail.com

## **Extended Abstract**

*Colour Etude I* is the first in a series of pieces exploring the strict application of certain spectral techniques influenced by the research of William Sethares, while keeping all parameters except for harmony very simple. These pieces function as a means to test harmonic concepts, and to create more accessible examples of microtonal and spectral music by leaving most other aspects of the piece uncomplicated.

“The more I experimented with alternative tunings, the more it appeared that certain kinds of scales sound good with some timbres and not with others. Certain kinds of timbres sound good in some scales and not in others.”<sup>1</sup> This quote from William A. Sethares stems from the beginnings of his research in Psychoacoustics and sensory consonance. His research would go on to suggest that there is a correlation between the pitches which occur as partials in a sound's spectrum, and the pitches which create the scale affording the most perceived consonance when writing music using that sound.

This had implications about harmony that interested me greatly, and I've applied aspects of this theory to several pieces of mine in many different ways. Some of the intended results of doing this have included: creating unusual microtonal modes which are spectrally derived; developing a kind of structurally functional harmony using these modes; attempting control perceived consonance and dissonance; effectively blending electronic and acoustic sounds together within a single texture; and deriving multiple elements of a work, such as harmony and form, from a single source, usually an individual sound.

The ultimate purpose of all of the listed aims was to create a sound-world which was cohesive, yet unique and distinctive to each piece, while still allowing for music with contrast and an engaging dramatic shape. Nonetheless, the questions still remained: to what extent is the use of Sethares's theories actually contributing to this final goal, and can his assertions actually be applied in a meaningful way within the context of a composition?

The difficulty in answering these questions lies in part in a lack of controlled examples clearly demonstrating the application Sethares's research. For example, the vast majority of music is made using sounds with varying spectra, not just a single spectrum which transposes in exact ratios as the fundamental changes; even a piece for solo instrument will demonstrate spectral variety in different registers. Sethares himself has written many examples that do effectively demonstrate his point, but the problem is that, from a musical point of view, they simply don't sound very good; this due largely to the demonstrative function of these examples, rather than their being true performance-designed compositions, and also because the sound design in question is very old-fashioned and uninspiring.

I wanted to put the effectiveness of Sethares's results to the test by writing a piece which puts them in the absolute foreground. Using additive synthesis, I could design a spectrum which could then be transposed exactly to reduce the kind of spectral variance previously discussed, and test whether I could compose an effective work using just one set of frequency ratios. To add to the emphasis on harmony, I wanted the piece to be very basic in all other regards; the form ended up as

essentially a simple theme and variation, for example, and there is little in the way of rhythmic development.

To begin with, an 11-partial waveform was generated in SuperCollider using random numbers for the partial frequencies and amplitudes, which I derived using SuperCollider's `ExpRand` method. The amplitudes were then tweaked to taste, but the frequencies were left as generated. The resulting ratios of the partials were as follows, where  $f$  is the fundamental frequency, to the nearest two decimal places:  $f$ ,  $1.87f$ ,  $1.97f$ ,  $2.10f$ ,  $3.20f$ ,  $3.40f$ ,  $5.24f$ ,  $7.00f$ ,  $8.61f$ ,  $8.92f$ ,  $9.97f$ .

The entire piece uses instances of waveforms with these same ratios between the frequencies of their 11 partials - i.e. exact transpositions of the same waveform. Rather than deriving modes from these frequencies, the exact pitches are used only, while a section is centered on a particular fundamental; to put it into terms relating to tonal music, in a particular 'key', only notes whose fundamental frequencies occur in the spectrum of the 'tonic' can be used. This leads to a series of 11-pitch sets which modulate between each other via common pitches, and the modulations are the structural signposts of the piece.

## References

- [1] Sethares, William A. *Tuning, Timbre, Spectrum, Scale*. London: Springer-Verlag, 1998.

## *Desdobramentos do contínuo for violoncello and live electronics*

Danilo Rossetti<sup>1</sup>, William Teixeira<sup>2</sup>

<sup>1</sup> First Laboratory – Interdisciplinary Nucleus for Sound Communication UNICAMP, Rua da Reitoria, 165 – “Cidade Universitária Zeferino Vaz” – 13083-872 Campinas, SP

<sup>2</sup> Second Laboratory, Federal University of Mato Grosso do Sul Avenida Costa e Silva, 79070-900, Campo Grande, MS

danilo.rossetti@nics.unicamp.br, william.teixeira@ufms.br

*Desdobramentos do contínuo* is a work for violoncello and live electronics written in 2016. It was the last live electroacoustic music composed during Rossetti's Ph.D. thesis [1], defended in the same year. The work was composed in collaboration with the violoncellist William Teixeira, for whom it is dedicated.

In terms of structuration, the piece involves two types of electroacoustic procedures, in addition to the instrumental writing of the score. Among the electroacoustic procedures, there are tape sounds, pre-composed in a studio in deferred time [2] and real-time treatments applied to the violoncello sound captured live.

For the composition of the deferred time sounds (tapes), time-frequency analysis procedures of phase vocoder [3] and convolution [4] were employed. From them, pitch-shifting and time-stretching processes could be implemented. Both of these procedures were performed having violoncello recordings as source material.

For the real-time processing, which was implemented in Max MSP, objects of HOA (High Order Ambisonics) Library – developed by the CICM (*Centre de recherche Informatique et Création Musicale*) of Université Paris 8 – were used. Employing the process object belonging to this library it is possible to combine electroacoustic treatments such as granulation, convolution, dephasing and microtemporal decorrelation (among others), with a high order ambisonics spatialization [5].

From a morphologic standpoint, the composed tape sounds have a continuous development in time, while the real-time generated sounds (mainly produced by a granulation process) have a discontinuous structure.

The motivation for this composition comes from an interpretation of René Thom's Catastrophe Theory [6], explained by him as a methodology that allows the organization of experience information in various conditions. Thom searched to describe discontinuities observed in a system evolution. He assumed that a system is presented as a succession of continuous evolutions, separated by abrupt leaps of qualitatively different natures.

We imagine the morphology of the whole piece as a continuity formed by, in its interior, continuous and discontinuous instrumental and electroacoustic sounds which are overlapped. This superposition of different sounds produces the timbre of the piece that can be perceived as one unity of form.

The objective of combining and overlapping deferred time sounds and real-time electroacoustic treatments to the violoncello acoustical sound is to create different layers that would merge together into a single structure. Our hypothesis in relation to both electroacoustic sounds (represented by continuous and discontinuous structures) is that this overlapping would be complementary and permeable in terms of sound morphology, generating a single perceived timbre.

In relation to the dialogue between instrumental and electroacoustic sounds, concepts of interaction and convergence [7] were addressed. Interaction is defined as a kind of action that occurs as two or more objects have an effect one upon another, where the idea of a two-way effect is essential. Convergence can be explained as the movement towards union or uniformity, the independent development of similar characters associated with similarity of an environment, and the merging of distinct technologies or devices into



a unified whole.

As a result of this dialogue, the amplification of the violoncello instrumental possibilities (in terms of information) is expected, and also the creation of new sonorities and transients.

This art submission is related to SBCM 2017 Music Paper “An Analysis of *Desdobramentos do Continuo* for Violoncello and Live Electronics Performed by Audio Descriptors”.

Mechanics and Acoustics, 2016.

## References

[1] Danilo Rossetti. Processos microtemporais de criação sonora, percepção e modulação da forma: uma abordagem analítica e composicional. Tese de Doutorado. Instituto de Artes, Universidade Estadual de Campinas, Campinas, 2016.

[2] Flo Menezes. Fusão e contraste entre a escritura instrumental e as estruturas eletroacústicas. In Flo Menezes, *Atualidade estética da música eletroacústica*, Editora UNESP, pages 13-20, 1999.

[3] Richard Dolson. The Phase Vocoder: A Tutorial. In *Computer Music Journal* v. 10, nº 4, 1986, pages 14-27.

[4] Curtis Roads. Musical Sound Transformation by Convolution. In *Proceedings of ICMC 1993*, pages 102-109. The International Computer Music Association, 1993.

[5] Pierre Gillot. Documentation de la bibliothèque HOA: Les champs sonores. CICM, Université Paris 8, Labex Arts H2H. <http://www.mshparisnord.fr/hoalibrary/ambisonie/les-champs-sonores/>

[6] René Thom. *Parábolas e catástrofes*. Entrevista sobre matemática, ciência e filosofia conduzida por Giulio Giorello e Simona Morini. Lisboa: Don Quixote, 1985.

[7] Danilo Rossetti. Interaction, Convergence and Instrumental Synthesis in Live Electronic Music. In: M. Aramaki, R. Kronland-Martinet, S. Ystad, editors, *Proceedings of the 12<sup>th</sup> CMMR*, pages 209-216, The Laboratory of

**Name:** Sever Tipei, Computer Music Project, University of Illinois, [s-tipei@illinois.edu](mailto:s-tipei@illinois.edu)

**Title:** figer, for fixed media

**Program Notes:** **figer**(*fr.*), *vb.* to clot, coagulate, congeal. Realized with DISSCO, software for Computer-assisted Composition and Additive Sound Synthesis developed at the UIUC Computer Music Project and Argonne National Laboratory, **figer** contains elements of indeterminacy at all structural levels. As such, it is a *composition class* or a *manifold composition*: all its actual and potential variants share the same structure, but differ in the way events are arranged in time and details are crafted.

The work includes four disquieting sections, three interludes and a coda; together they suggest an “apocalyptic” picture of surrealist aural images. Similar to paintings born from that aesthetics, it includes recognizable, familiar elements placed in an incongruent context. The coda, a quote from the traditional repertoire, enforces this perception.

In **figer** there are no themes-characters participating in a logical “plot”. Instead various sound objects re-occur in a non-linear, ostensibly random succession and listeners are invited to create their own representations of the proposed sound shapes. New appearances of previously encountered entities are distinct although they all can be identified as being incarnations of three primary types of materials: points, lines, and aggregates/chords.

The work could also be seen as a riddle, the answer to be found in the coda and the title.

**Relates to paper:** submission 170964, Sever Tipei - “Communicating a World View: figer, a Manifold Composition”.

**Venue:** evening concert, stereo and 8 channel versions available

**Duration:** 12 min.

**Instrumentation:** fixed media, stereo and 8 channel versions available

**Composer bio:** Sever Tipei was born in Bucharest, Romania, and immigrated to the United States in 1972. He holds degrees in composition and piano performance from the University of Michigan and Bucharest Conservatory. Tipei has taught Composition and Music Theory since 1978 at the University of Illinois at Urbana-Champaign School of Music where he directs the Computer Music Project; he is also a National Center for Supercomputing Applications (NCSA) Faculty Affiliate. Between 1993 and 2003 Tipei was a Visiting Scientist at Argonne National Laboratory where he worked with mathematician Kans G. Kaper on sonification of complex scientific data.

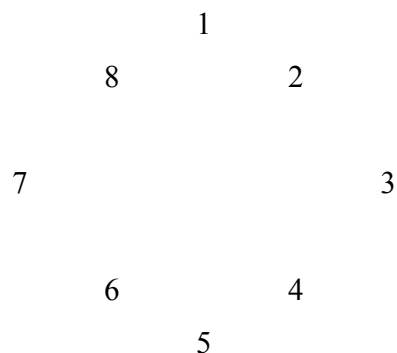
Sever Tipei's main fields of interest are Computer Music and Music Formalization. Most of his

compositions were produced with software he designed: MP1 - for Computer-assisted Composition; DIASS and DISCO - programs for sound synthesis; and M4CAVE - for the visualization of music in an immersive virtual environment. MP1, first used in 1973, was the first such program to run on a supercomputer (CRAY-XMP, in 1986). More recently, Tipei and his collaborators have developed DISSCO, software that unifies Computer-assisted (Algorithmic) Composition and Sound Synthesis into a seamless process.

Sever Tipei's papers have appeared in the Computer Music Journal, Leonardo, and in the proceedings of various International Computer Music Conferences, Music Perception and Cognition, WSEAS, and others. In 1989 he introduced the concept of *manifold composition*, the collection of all actual and potential variants of a computer generated musical work which contains elements of indeterminacy. As a pianist, he has performed in the United States, Korea, France, Italy, Belgium, the Netherlands, and Romania, and recorded for the ORION label.

Tipei regards the computer as a collaborator whose skills and abilities complement those of the human artist. He sees the composition of music both as an experimental and a speculative endeavor that delivers a particular world view.

**Technical requirements:** either stereo or 8 channel playback system. If 8 channels, the tracks are arranged as follows:



**Audio file at:** <https://soundcloud.com/sever-tipei/figer-1>

# For alto saxophone and live electronic sounds: “era como se estivéssemos vivos”

Rodolfo A. D. V. Valente

Escola de Comunicações e Artes da Universidade de São Paulo  
Av. Prof. Lúcio Martins Rodrigues, 443 – 05508-020 São Paulo, SP

rodolfovalente@usp.br, rodolfo@rodolfovalente.com

## Extended Abstract

Scored for amplified alto saxophone and a live electronic interactive system, “era como se estivéssemos vivos” (translatable as “it felt as if we were alive”) takes advantage of the SuperCollider programming environment as a sound processing and live performance tool with special interest in its flexible machine listening capabilities [1] as well as implementations of physical models.

Exploring a highly energetic and gestural idiom, this piece brings forth more percussive and noisy characteristics of the instrument, inhabiting mostly the realm known as extended techniques and leaving smaller space for more conventionally intonated pitched materials. Thus the relationship between electronic and acoustic sounds is intended to reinforce the richness in articulation chosen for the instrumental writing. As a strategy to providing less linear yet expressive responsiveness from the electronic system, physical models of springs and bouncing objects [2], are implemented to control modulation envelopes and delay lines, which are activated by both direct audio input as well as an amplitude tracker at control rate.

A “ghost tape” constantly playing at zero amplitude, being brought into sounding when a certain threshold is hit, thus revealing underlying sonic layers created in real time by intensive manipulation of pre-recorded saxophone sounds. Further sonic complexity is generated by the second layer of live electronic processing. Granulated echoes implementing the behavior of physical models and

polyphonically chopped reverb tails running through independently variable filters, scatter and disembodied instrumental gestures in the performative space (which can be quickly redistributed for an array of 2 to 8 speakers, changing a single variable in the beginning of the code).

When pitched notes at last are allowed to come into play, ring modulation effects are carefully applied to distort the instrumental sound, producing “virtual multi-phonics”, which simultaneously expanding the work’s pitch structure and bringing electronic sounds closer with the already established extended technique vocabulary of the instrument.

## Download link for artistic submission info and score:

<https://drive.google.com/file/d/0B5xZVMvjv3hHYVpJTGJJR0FPZVk/view?usp=sharing>

## References

- [1] Nick Collins. Machine Listening in SuperCollider. In S. Wilson, D. Cottle, N. Collins, editors, *The SuperCollider Book*, pages 439-462. Massachusetts Institute of Technology, 2011.
- [2] Julius O. Smith. *Physical Audio Signal Processing*. <http://ccrma.stanford.edu/~jos/pasp/> online book, 2010 edition, accessed 29<sup>th</sup> May 2017.

# Lignes et Pointes - étude pour la décomposition en deux parties d'une oeuvre de Joan Miró

Antonio D'Amato

antonio089@yahoo.it

## Abstract

This piece comes from a research about the possibility of maximizing the transmission of emotions through a synesthetic transposition of abstract paintings into music. In fact from that point of view synesthesia is an interaction of different sensory modalities, assumed that in certain conditions a single sense could activate the others. On this side working on figurative artworks conceived in the 20th century could be a good testbed because the theme of the synesthesia between figurative arts and music has often involved artists of different movements.

The piece is a personal transposition into music of a gouache included in the first set of Constellations by Joan Miró, chosen by reason of a deep and personal study on chromatic relationships and of abstract geometrical forms conducted by Miró himself. The piece is formally conceived in two parts, intended as an etude on simple elements, grouped into two basic categories, each part focusing on different families of graphical elements. Long and slow elements are exclusively dominant in the first part, while impulsive sounds build up the second part. These elements are selected and extensively overlapped in order to develop an abstract study on basic elements of a music vocabulary.

Here the goal is to attempt forcing the merging of communicative strengths from different art forms through the extraction of the overall shape of each graphical element and a successive superimposition of their general traits to sound elements through intensive dsp.

The composition process involved two separate approaches in order to create basic sound elements to be overlapped and sequenced in the two sections. In the first section I used a software that allowed to transform images in sound textures, based on luminance factor and pixel location in the selected frame. It could be considered a noise-based synthesis, since it generates original sounds from images as output. It was not a mechanical process because it was clearly possible to experiment variants adopting as source material images modified with any image-processing software as the well known Photoshop or similar ones. Moreover, it was possible to limit the bandwidth of the noise-shaping process, or even adopt a linear or logarithmic scale in the location/pitch conversion. Consequently, a wide range of different results is possible. A further option in some cases involved a reverse process converting wildly-transformed sounds into images again, then processing them again as images and turning them back finally into the audio domain. These basic elements were processed with other DSP algorithms, mainly spectral delays and convolution with recorded string chords. In the second section I used a collection of heterogeneous impulsive sounds, synthetic or recorded, and I displaced them on a regular time grid, in pattern. Each pattern was somehow derived on distances measured between *black spots* that the artist painted sparsely on the white paper. The dimension of the spots was assumed as intensity cue. After that the patterns were elaborated with the classic canonic imitation procedures: inversion, contrary motion and the combination of both.

## Puzzle Pieces

Dr. Paul Schuette, DMA  
University of the Arts  
e: [pschuette@uarts.edu](mailto:pschuette@uarts.edu)  
p: 1+ (312) 320-8663

### Program Note

*Puzzle Pieces* was commissioned by The Stockhausen Response Project for pianist Brianna Madske

As a composer of electroacoustic music, the figure of Stockhausen - the indelible German (or Sirius-ian?) explorer, technician, and mystic philosopher of 20th century music - looms large. In *Mikrophonie I* (the specific work that we were asked to respond to), Stockhausen breaks ground that the medium of electroacoustic music has in some ways been responding to ever since. From a technical standpoint, as one might surmise from the title, this work elevated the status of the microphone from a passive piece of hardware to an instrument capable of an extremely subtle range of expressive gestures. In fact to perform the work, one must become something of a virtuoso microphone performer in order to execute Stockhausen's incredibly detailed notation for the instrument. This perceptive restructuring liberated the status of electronics in music by putting the 'microphonist' on the same plane as the violinist. From this perspective, all of my music, which seeks to integrate electronics in nuanced and novel ways in order to enhance the range of expressive possibilities, is made possible by Stockhausen's contributions.

*Mikrophonie I* is also a primary example of another of Stockhausen's influential ideas: moment form. Simply put, Stockhausen's conception of a moment form is one in which, "no developmental direction can be predicted with certainty from the present one." Far from a license for piecemeal composition, Stockhausen was searching for a means to restructure the dimensions of music. By calling our attention to the 'Now', he seeks to, "make vertical slices, as it were, that cut through a horizontal temporal conception to a timelessness I call eternity: an eternity that does not begin at the end of time but is attainable in every moment. I am speaking of musical forms in which apparently nothing less is being attempted than to explode (even to overthrow) the temporal concept." By seeking to expand upon the dimensional planes in which the structural logic of the piece is projected, *Puzzle Pieces* is my humble attempt to expand upon the implications of Stockhausen's 'Now'.

Intended Venue: Evening concert

Duration: 13'00"

Instrumentation: Piano and live electronics

Performer: Dr. Brianna Matzke, DMA (can attend)

### Paul Schuette - Bio

Paul Schuette is a composer, sound artist, improviser, programmer, and educator living and working in Philadelphia, PA. According to Citybeat Cincinnati, he creates "works of art that address multiple senses simultaneously and thoughtfully, no matter the context." His work and pedagogy explore the potential for technology to enhance art and music in meaningful ways. Interested in writing complex music with simple resources, he describes his current compositional aesthetic as 'simple complexity'. Through the use of simple materials, he hopes to make complex forms accessible by presenting listeners with conceptions that they can see, remember, manipulate, and, ultimately, 'understand' by merely listening. Paul's gallery work, including his collaboration with painter Mary Laube (aka The Warp Whistle Project) has been exhibited in Chicago, Cincinnati, Bloomington-Normal, Detroit, and Daejeon, Korea. Recent musical collaborators include Percussion Group Cincinnati, Quinn Collins, Brianna Matzke, Erica Dicker, Zach Larabee, and Eric Derr. Paul has been a resident artist at VCCA, the Ucross Foundation, Signal Culture, and the Experimental Sound Studio. His music has been performed at numerous universities and at venues including the Contemporary Arts Center (Cincinnati), Loughheed-Kofoed Festival of the Arts, PASIC, Constellations (Chicago), New Music Gathering, NIME, SEAMUS, Cincinnati Fringe Festival, NYC Electroacoustic Music Festival, Soundcrawl:Nashville, Intermedia Arts Festival (Indianapolis), Taff's Art Center (Columbia, SC), and the Midwest Composer's Forum, among others.

### Brianna Matzke - Bio

Dr. Brianna Matzke's dynamic pianism shows "a sense of refinement, flair, and technical prowess" (clevelandclassical.com).. An avid performer and commissioner of new music, she has collaborated with many composers, including Michael Fiday, Elliot Cole, Marc Mellits, Mark Mothersbaugh, Douglas Knehans, Molly Joyce, Alexandra Du Bois, D. J. Sparr, Nate May, Tyler Eschendahl, Dylan Sheridan, Stephanie Ann Boyd, Paul Schuette, Danny Clay, Jennifer Jolley, Carrie Magin, Evan Williams, Paul Poston, Bryan Percoco, Trevor Gomes, and Lindsey Jacob.

Her ongoing commissioning initiative, called The Response Project, has premiered five new works for solo piano and electronics written in response to *Microphonie I* by Karlheinz Stockhausen, and will present five new works for violin and piano in 2017, written in response to the phrase, "on behalf."

In addition to performing, Brianna is a dedicated music educator and pedagogue. A Nationally Certified Teacher of Music (NCTM), she has served on the faculties of the Oberlin Conservatory, Interlochen Arts Camp, Wilmington College, Thomas More College, and the University of Cincinnati College-Conservatory of Music Preparatory Department. As an educator, Brianna believes in the power of music to incite positive social change, and she encourages that change by serving as President of the Ohio Music Teachers Association Southwest Division.

She holds degrees from the University of Cincinnati College-Conservatory of Music (CCM) and the University of Kansas.

**Technical Requirements:**

- stereo playback; microphone on piano running through Max/MSP
- can provide laptop, interface, and MIDI footpedal
- detailed plot contained in score

**Setup Time:** 30' - including soundcheck

**Stage Plot:** see score for details



## Puzzle Pieces - Program Note

Dr. Paul Schuette, DMA <sup>1</sup>

<sup>1</sup> University of the Arts, Philadelphia, PA, USA

320 S Broad St - 19102

pschuette@uarts.edu

### Abstract

humble attempt to expand upon the implications of Stockhausen's 'Now'.

*Puzzle Pieces* was commissioned by The Stockhausen Response Project for pianist Brianna Matzke

As a composer of electroacoustic music, the figure of Stockhausen - the indelible German (or Siriusian?) explorer, technician, and mystic philosopher of 20th century music - looms large. In *Mikrophonie I* (the specific work that we were asked to respond to), Stockhausen breaks ground that the medium of electroacoustic music has in some ways been responding to ever since. From a technical standpoint, as one might surmise from the title, this work elevated the status of the microphone from a passive piece of hardware to an instrument capable of an extremely subtle range of expressive gestures. In fact to perform the work, one must become something of a virtuoso microphone performer in order to execute Stockhausen's incredibly detailed notation for the instrument. This perceptive restructuring liberated the status of electronics in music by putting the 'microphonist' on the same plane as the violinist. From this perspective, all of my music, which seeks to integrate electronics in nuanced and novel ways in order to enhance the range of expressive possibilities, is made possible by Stockhausen's contributions.

*Mikrophonie I* is also a primary example of another of Stockhausen's influential ideas: moment form. Simply put, Stockhausen's conception of a moment form is one in which, "no developmental direction can be predicted with certainty from the present one." Far from a license for piecemeal composition, Stockhausen was searching for a means to restructure the dimensions of music. By calling our attention to the 'Now', he seeks to, "make vertical slices, as it were, that cut through a horizontal temporal conception to a timelessness I call eternity: an eternity that does not begin at the end of time but is attainable in every moment. I am speaking of musical forms in which apparently nothing less is being attempted than to explode (even to overthrow) the temporal concept." By seeking to expand upon the dimensional planes in which the structural logic of the piece is projected, *Puzzle Pieces* is my

# Puzzle Pieces

for solo piano and electronics

Paul Schuette



## PERFORMANCE NOTES

### Electronics

- The performer should be miked with (at least) a stereo pair of microphones. The sound designer may use more microphones at their discretion.
- A computer equipped with Max/MSP is required to run the audio program. Contact the publisher at [www.paulschuette.com](http://www.paulschuette.com) for the required patches.
- An audio interface which can facilitate 2 XLR inputs and 2 separate output channels is required
- A MIDI device which enables the use of a foot pedal is also required

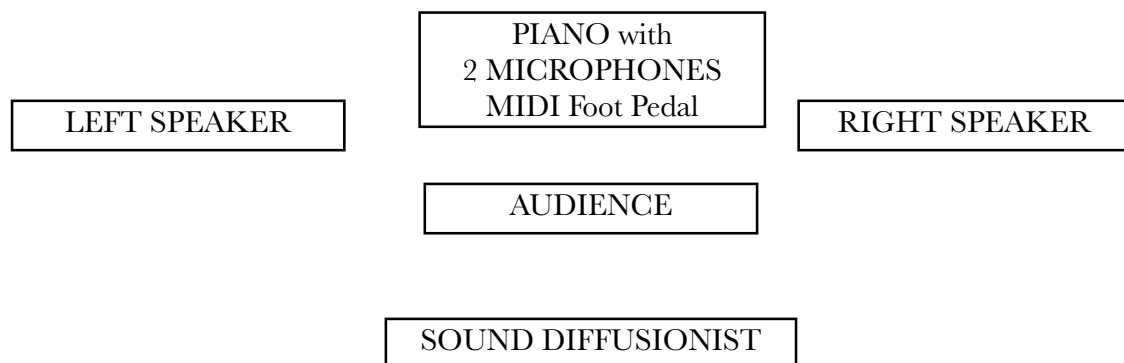
### MIDI Pedal Notation and Use

- Each section of the piece corresponds to a specific setting in the patch indicated by the number above the cue.
- Moving between these settings is cued via the MIDI foot pedal
- While notated at the start of each section, the foot pedal should be struck in advance of each section as an ‘eight-note-pickup’ or quasi grace note prior to the downbeat of each section.

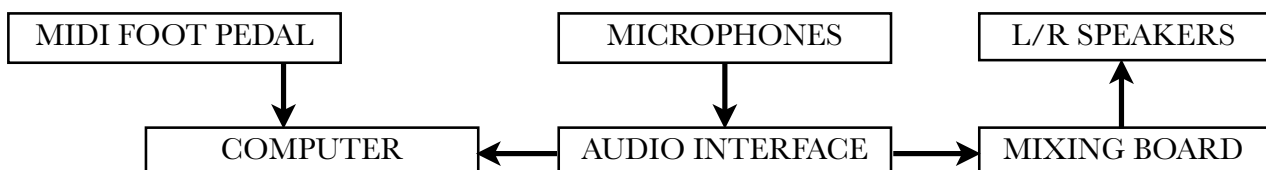
### Notation

- Cesuras should be thought of as longer pauses than breath marks - cesuras lasting between 1-3” and breath marks for a beat or half beat in the given tempo.
- Clusters, indicated by block style note-heads, should consist of all black and white keys in the approximate range.

## SETUP



## SIGNAL ROUTING



## PROGRAM NOTES

*Puzzle Pieces* was commissioned by The Stockhausen Response Project for pianist Brianna Madske

As a composer of electroacoustic music, the figure of Stockhausen - the indelible German (or Sirius-ian?) explorer, technician, and mystic philosopher of 20th century music - looms large. In *Mikrophonie I* (the specific work that we were asked to respond to), Stockhausen breaks ground that the medium of electroacoustic music has in some ways been responding to ever since. From a technical standpoint, as one might surmise from the title, this work elevated the status of the microphone from a passive piece of hardware to an instrument capable of an extremely subtle range of expressive gestures. In fact to perform the work, one must become something of a virtuoso microphone performer in order to execute Stockhausen's incredibly detailed notation for the instrument. This perceptive restructuring liberated the status of electronics in music by putting the 'microphonist' on the same plane as the violinist. From this perspective, all of my music, which seeks to integrate electronics in nuanced and novel ways in order to enhance the range of expressive possibilities, is made possible by Stockhausen's contributions.

*Mikrophonie I* is also a primary example of another of Stockhausen's influential ideas: moment form. Simply put, Stockhausen's conception of a moment form is one in which, "no developmental direction can be predicted with certainty from the present one." Far from a license for piecemeal composition, Stockhausen was searching for a means to restructure the dimensions of music. By calling our attention to the 'Now', he seeks to, "make vertical slices, as it were, that cut through a horizontal temporal conception to a timelessness I call eternity: an eternity that does not begin at the end of time but is attainable in every moment. I am speaking of musical forms in which apparently nothing less is being attempted than to explode (even to overthrow) the temporal concept." By seeking to expand upon the dimensional planes in which the structural logic of the piece is projected, *Puzzle Pieces* is my humble attempt to expand upon the implications of Stockhausen's 'Now'.

# Puzzle Pieces

Score

*for Brianna Madske*

Paul Schuette

♩ = 108, **punchy**

Piano

MIDI Pedal

\* B/W cluster - range approx.

Puzzle Pieces

26

**raucous** \* B/W cluster - range approx.

30

**A** ♩ = 66, woozy

34

38

**rubato**

41

Puzzle Pieces

Musical score for measures 45-56. The piece is in 5/16 time. The score features a piano part with triplets and dynamic markings of *fff* and *ppp*. The right hand has a melodic line with triplets, while the left hand provides a rhythmic accompaniment with triplets. The piece concludes with a double bar line and a 5/16 time signature.

**B** ♩ = 108, precise - delicate

Musical score for measures 57-63. The tempo is marked *p*. The score is in 5/16 time and features a piano part with a 5-finger exercise labeled "5 una corde (to C)". The right hand has a melodic line with slurs, and the left hand has a rhythmic accompaniment. The piece concludes with a double bar line and a 5/16 time signature.

Musical score for measures 64-72. The score is in 3/8 time and features a piano part with a forte dynamic marking of *f*. The right hand has a melodic line with slurs, and the left hand has a rhythmic accompaniment. The piece concludes with a double bar line and a 5/16 time signature.

Musical score for measures 73-81. The tempo is marked *rit. (on repeat)* with a tempo of ♩ = 66. The score is in 5/16 time and features a piano part with a piano dynamic marking of *p*. The right hand has a melodic line with slurs, and the left hand has a rhythmic accompaniment. The piece concludes with a double bar line and a 5/16 time signature.

Musical score for measures 82-90. The tempo is marked *accel.*. The score is in 5/16 time and features a piano part with a piano dynamic marking of *p*. The right hand has a melodic line with slurs, and the left hand has a rhythmic accompaniment. The piece concludes with a double bar line and a 5/16 time signature.



## Puzzle Pieces

83

89 *- a tempo*

95

98 *rit.*

107  $(\text{♩} = 66)$

Puzzle Pieces

*accel.*

*a tempo* C // ♩ = 66, majestic

*f* *ff*

*Ped.* *tre corde*

*8va* ♩ = 108, digitally

*mf* *ff* *mf* *ff*

*7*

*8va*

*mf* *p* *f* *p* *mf*

*8vb*

*8va*

*ff* *p*

*8vb*

6

Puzzle Pieces

Musical score for measures 143-146. The piece is in 4/4 time. Measure 143 starts with a treble clef, a key signature of one sharp (F#), and a dynamic marking of *f*. The right hand plays a series of eighth notes, while the left hand is silent. In measure 144, the right hand continues with eighth notes, and the left hand enters with a bass clef, playing a rhythmic pattern of eighth notes. Dynamic markings *p* and *ff* are present. Measure 145 shows the right hand with a *p* dynamic and the left hand with a *ff* dynamic. Measure 146 ends with a *p* dynamic in the right hand.

Musical score for measures 147-150. Measure 147 features a treble clef, a key signature of one sharp (F#), and a dynamic marking of *ff*. The right hand has a *8va* marking above it. The left hand plays a rhythmic pattern of eighth notes. Measure 148 continues with the left hand's pattern and a *p* dynamic in the right hand. Measure 149 shows the left hand's pattern and a *p* dynamic in the right hand. Measure 150 ends with a *p* dynamic in the right hand.

Musical score for measures 151-154. Measure 151 starts with a treble clef, a key signature of one sharp (F#), and a dynamic marking of *f*. The right hand is silent, while the left hand plays a rhythmic pattern of eighth notes. Measure 152 shows the right hand with a *p* dynamic and the left hand with a *mf* dynamic. Measure 153 features a *ff* dynamic in the right hand and a *p* dynamic in the left hand. Measure 154 ends with a *p* dynamic in the right hand.

Musical score for measures 155-158. Measure 155 starts with a treble clef, a key signature of one sharp (F#), and a dynamic marking of *ff*. The right hand is silent, while the left hand plays a rhythmic pattern of eighth notes. Measure 156 shows the right hand with a *ff* dynamic and the left hand with a *p* dynamic. Measure 157 features a *ff* dynamic in the right hand and a *p* dynamic in the left hand. Measure 158 ends with a *p* dynamic in the right hand.

Musical score for measures 159-162. Measure 159 starts with a treble clef, a key signature of one sharp (F#), and a dynamic marking of *ff*. The right hand plays a series of eighth notes, while the left hand is silent. Measure 160 shows the right hand with a *p* dynamic and the left hand with a *ff* dynamic. Measure 161 features a *ff* dynamic in the right hand and a *mf* dynamic in the left hand. Measure 162 ends with a *mf* dynamic in the right hand. A box labeled 'D' is present above measure 161. A tempo marking of  $\text{♩} = 66$  is shown. A pedal marking 'Ped.' is present below measure 162. A page number '8' is at the bottom right.

Puzzle Pieces

164

*p*

Ped.

9

173

*p* *ff* *p*

10

177

180

184

Puzzle Pieces

Musical score for measures 187-188. The piece is titled "Puzzle Pieces". It features a piano (p) section with triplets in both hands, transitioning to a fortissimo (ff) section. The notation includes various rhythmic patterns and dynamic markings.

♩ = 108

Musical score for measures 189-191. The tempo is marked as ♩ = 108. The piece continues with fortissimo (ff) dynamics and complex rhythmic patterns, including triplets and slurs.

Musical score for measures 192-195. The piece features a piano (p) section followed by a fortissimo (ff) section. The notation includes various rhythmic patterns and dynamic markings.

Musical score for measures 196-199. The piece continues with fortissimo (ff) dynamics and complex rhythmic patterns, including triplets and slurs.

F ♩ = 66 freely

D# *molto rit.*

E

p

E

ppp

12

Musical score for measures 200-203. The piece features a fortissimo (ff) section with a tempo of ♩ = 66, marked "freely". It includes a section with a D# chord and a "molto rit." marking, followed by a piano (p) section and a section with a D# chord and a "ppp" marking. The piece concludes with a final chord and a "12" marking.

**♩ = 108, digitally** Puzzle Pieces

207 *mf* *ff* *p* *ff*

13

213 *mf* *ff*

219 *p* *ff* *p*

223 *p*

227 *ff* *p* *ff*

Puzzle Pieces

233

Musical score for measures 233-239. The right hand features a melodic line with accents and slurs, while the left hand has a bass line with slurs and accents. Dynamics include *p*, *ff*, and *p*.

240

Musical score for measures 240-247. The right hand contains triplets and slurs, with dynamics *ff*, *mf*, and *p*. The left hand has a bass line with slurs and dynamics *p* and *ff*. Octave markings *8va* and *8vb* are present.

248

Musical score for measures 248-255. The right hand has a complex melodic line with slurs and dynamics *p* and *ff*. The left hand has a bass line with slurs and dynamics *p*.

256

Musical score for measures 256-259. The right hand has a melodic line with slurs and dynamics *p*. The left hand has a bass line with slurs and dynamics *p*.

260

Musical score for measures 260-267. The right hand has a melodic line with slurs and dynamics *ff*, *p*, and *ff*. The left hand has a bass line with slurs and dynamics *ff*, *p*, and *ff*. A triplet is marked in the right hand.

Puzzle Pieces

Musical score for measures 265-270. The system includes a grand staff with treble and bass clefs. Measure 265 features a treble clef with eighth-note patterns and a bass clef with a whole note. Measures 266-267 show treble clef with eighth-note patterns and bass clef with whole notes. Measure 268 has a treble clef with eighth-note patterns and a bass clef with a whole note. Measure 269 has a treble clef with eighth-note patterns and a bass clef with a whole note. Measure 270 has a treble clef with eighth-note patterns and a bass clef with a whole note. A dynamic marking of *p* is present in measure 270. A box labeled 'G' is above measure 270, and a tempo marking of  $\text{♩} = 66$  is also present. An 8va marking is at the end of the system.

Musical score for measures 270-275. The system includes a grand staff with treble and bass clefs. Measure 270 has a treble clef with eighth-note patterns and a bass clef with a whole note. Measure 271 has a treble clef with eighth-note patterns and a bass clef with a whole note. Measure 272 has a treble clef with eighth-note patterns and a bass clef with a whole note. Measure 273 has a treble clef with eighth-note patterns and a bass clef with a whole note. Measure 274 has a treble clef with eighth-note patterns and a bass clef with a whole note. Measure 275 has a treble clef with eighth-note patterns and a bass clef with a whole note. A dynamic marking of *p* is present in measure 270, *ff* in measure 271, and *p* in measure 272. A box labeled 'G' is above measure 270, and a tempo marking of  $\text{♩} = 66$  is also present. A Ped. marking is in measure 272. An 8va marking is at the end of the system.

Musical score for measures 275-280. The system includes a grand staff with treble and bass clefs. Measure 275 has a treble clef with eighth-note patterns and a bass clef with a whole note. Measure 276 has a treble clef with eighth-note patterns and a bass clef with a whole note. Measure 277 has a treble clef with eighth-note patterns and a bass clef with a whole note. Measure 278 has a treble clef with eighth-note patterns and a bass clef with a whole note. Measure 279 has a treble clef with eighth-note patterns and a bass clef with a whole note. Measure 280 has a treble clef with eighth-note patterns and a bass clef with a whole note. A dynamic marking of *ppp* is present in measure 275, and *mf* in measure 278. A box labeled '14' is below measure 275. The word 'rubato' is written above measure 275. The system ends with a double bar line and a 4/4 time signature.

Musical score for measures 280-283. The system includes a grand staff with treble and bass clefs. Measure 280 has a treble clef with eighth-note patterns and a bass clef with a whole note. Measure 281 has a treble clef with eighth-note patterns and a bass clef with a whole note. Measure 282 has a treble clef with eighth-note patterns and a bass clef with a whole note. Measure 283 has a treble clef with eighth-note patterns and a bass clef with a whole note. A dynamic marking of *pp* is present in measure 280, and *f* in measure 281. A box labeled '16' is below measure 280. The word 'accel.' is written above measure 280. A tempo marking of  $\text{♩} = 108, \text{punchy}$  is present. The system ends with a double bar line and a 4/4 time signature.

Musical score for measures 283-288. The system includes a grand staff with treble and bass clefs. Measure 283 has a treble clef with eighth-note patterns and a bass clef with a whole note. Measure 284 has a treble clef with eighth-note patterns and a bass clef with a whole note. Measure 285 has a treble clef with eighth-note patterns and a bass clef with a whole note. Measure 286 has a treble clef with eighth-note patterns and a bass clef with a whole note. Measure 287 has a treble clef with eighth-note patterns and a bass clef with a whole note. Measure 288 has a treble clef with eighth-note patterns and a bass clef with a whole note. The system ends with a double bar line and a 4/4 time signature.



Puzzle Pieces

Musical score for measures 286-288. The system consists of a grand staff with a treble clef on the upper staff and a bass clef on the lower staff. Measure 286 features a treble staff with eighth-note triplets and a bass staff with eighth-note triplets. Measures 287 and 288 continue with similar rhythmic patterns.

Musical score for measures 289-292. Measure 289 has eighth-note triplets in both staves. Measure 290 includes a double bar line and a repeat sign. Measures 291 and 292 feature a dense texture of chords in the treble staff, with the label *8va* above the staff, and eighth-note patterns in the bass staff, with the label *8vb* below the staff.

17

Musical score for measures 293-296. Measures 293 and 294 show dense chordal textures in the treble staff (*8va*) and eighth-note patterns in the bass staff (*8vb*). Measures 295 and 296 feature a mix of chords and eighth-note patterns in both staves.

Musical score for measures 297-302. Measures 297 and 298 have chords in the treble staff (*8va*) and eighth-note patterns in the bass staff (*8vb*). Measures 299 and 300 continue with similar textures. Measures 301 and 302 feature eighth-note triplets in both staves.

Musical score for measures 303-306. Measures 303 and 304 have dense chordal textures in the treble staff (*8va*) and eighth-note patterns in the bass staff (*8vb*). Measures 305 and 306 feature eighth-note triplets in both staves.

Puzzle Pieces

Musical score for measures 307-314. The score is in treble and bass clefs. The treble clef part features a sequence of chords and triplets, with dynamic markings *8<sup>va</sup>* and accents. The bass clef part features a sequence of chords and triplets, with dynamic markings *8<sup>vb</sup>* and accents. The piece concludes with a double bar line.

Musical score for measures 315-318. The score is in treble and bass clefs. The treble clef part features a sequence of chords and triplets, with dynamic markings *8<sup>va</sup>* and accents. The bass clef part features a sequence of chords and triplets, with dynamic markings *8<sup>vb</sup>* and accents. The piece concludes with a double bar line.

Cincinnati and Union Pier  
2014-15

- Author name: **Kyong Mee Choi**
- Affiliations (if relevant): **Roosevelt University**
- Contact information: [kchoi@roosevelt.edu](mailto:kchoi@roosevelt.edu) (1-773-910-7157)
- Title of the proposal: *rare yet soft*
- Program notes (1-3 paragraphs): *rare yet soft* explores the subtlety of quoted thematic material from Mahler's Symphony No. 5 Adagietto. The piece has three sections when each quote is introduced in a different context. At the end, the piece shows how subtle influence of this quotation can affect the overall shape of the piece. This piece is dedicated the composer's beloved father, Soon Bong Choi.
- Indication if the performance relates to a separate presentation in the technical program (in this case, please specify the title, authors and submission number of the separate submission): **N/A**
- Intended venue: evening concert, installation, etc.: **Concert**
- Duration and instrumentation: **7 min. 11 seconds** (no instrument)
- Names of performers, if applicable: **N/A**
- Composer / performer bios or a link to where these can be found online

[http://www.kyongmeechoi.com/Main\\_Site/Bio.html](http://www.kyongmeechoi.com/Main_Site/Bio.html)

# SBCM 2017

## Suíte [en]Quadrada

- KairosPania, Orquestra Errante & Luzilei Aliel -

Yonara Dantas<sup>1</sup>, Migue Antar<sup>2</sup>, Luzilei Aliel<sup>2\*</sup>

<sup>1</sup> Escola Superior de Artes Celia Helena, Av. São Gabriel, 462 - 01435-000 São Paulo, SP

<sup>2</sup> Núcleo de Pesquisas em Sonologia (NuSom) – Escola de Comunicações e Artes da Universidade de São Paulo. Av. Prof. Lúcio Martins Rodrigues, 443 – 05508-020 São Paulo, SP

[yoyodantas@gmail.com](mailto:yoyodantas@gmail.com), [miguedz5@gmail.com](mailto:miguedz5@gmail.com), [luzaliel@gmail.com](mailto:luzaliel@gmail.com)

### Abstract (Sinopse)

The performance develops in the wake of dialogues and frictions between three languages: free musical improvisation, acting and live electronics. The performance explores the narrativity imprinted on moving bodies, the visual force of live video recording of these same bodies (triggering the everyday life of the bifurcation between real life and life on the screen) and acoustic and electronic sounds amalgamating the performance. The *Suíte [en] Quadrada* is an achievement of the artistic collective KairosPania, Orquestra Errante & Luzilei Aliel.

### 1. Program notes

The *Suíte [en]Quadrada* is a variation of the television piece called "Quad" (Beckett, 1980). In this comprovisation (composition + musical improvisation: Aliel et al., 2015; Antar, 2016), we implemented a complex semi adaptive system to organize the scenic processes, free musical improvisation and real-time audio processing. In the *Suíte [en]Quadrada*, self-absorbed figures move in a mathematical way in a square. We use Beckett's work to construct a variation of the initial concept proposing the implementation of contingencies through free improvisation and visual and sound processing.

We aim at a structure of approximation of the three strands proposed in an interdisciplinary process capable of providing singular artistic unfolding. The boundaries between artistic languages are increasingly tenuous. Our research of this premise dialogue between free musical improvisation, acting and audio processing and sound in real time. We consider this type of research pertinent, after all, we do not find a massive academic research that examines alignments and conflicts between free improvisation, sound and visual processing with the performing arts.

Methodologically, we developed a socio-ecological system (SES), that is, a complex adaptive system that is characterized by self-organization and distributed control, (Sibertin-Blanc et al., 2011) for the *Suíte [en]Quadrada*. This system includes multiple agents with diverse and contrasting interests and management objectives, acting at different spatial and temporal levels. In our perspective, this methodology, based on processes of comprovisation, allows us to equate electronic processes, such as real-time sound processing, to artistic expression levels such as free musical improvisation and acting. In this way, the performance uses the capture of the movements of the actors via webcam and algorithms in Pure Data (PD) to process auditory and visual contents causing unfolding the initially acoustic events.

### 2. Intended venue

Auditorium with space for 17 performers. Preferably with Italian stage (non-exclusive).

---

\*Supported by Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) - (processo n° 2015/10342-1)

### 3. Duration and instrumentation

The *Suíte [en]Quadrada* performance lasts 20 minutes. It features: voice, saxophones, transverse flute, acoustic double bass, prepared guitars, acoustic piano, bass clarinet, percussion, luthier's experimental instruments and synthesized sounds (PD).

### 4. Names of performers (Technical file)

Suíte [en]Quadrada - *KairosPania, Orquestra Errante & Luzilei Aliel*

\* Composition: Yonara Dantas, Migue Antar e Luzilei Aliel

\* General direction: Yonara Dantas

\* Musical direction: Migue Antar

\* Live Eletronics: Luzilei Aliel

\* Actresses and actors of KairosPania: Anna Kobzareva, Carmen Estevez, Filipe Augusto, Kelly Caldas, Lara Kadocsa, Victor Pessoa e Yonara Dantas.

\* Musicians of Orquestra Errante: Ariane Stolfi, Caio Righi, Fabio Manzione, Fabio Martinelli, Inés Terra, Luzilei Aliel, Mariana Carvalho, Max Schenkman, Migue Antar, Natalia Francischini, Pedro Sollero, Rogério Costa.

\* Lighting: Alexandre Passos.

### 5. Composer/performer bios

Yonara Dantas: <http://lattes.cnpq.br/2886259247901984>

Migue Antar: <http://lattes.cnpq.br/0508526640066734>

Luzilei Aliel: <http://lattes.cnpq.br/9729924267239229>

### 6. Detailed description of technical requirements

The *Suíte [en]Quadrada* prioritizes the acoustic sound of the instruments. Amplification occurs only in a punctual way on prepared guitars, each of which has its own portable amplifier (small size). The other instruments sound acoustically, without amplification. Some instrumentalists are positioned on the stage, while others are seated next to the audience. The *Suíte [en]Quadrada* was planned for an auditorium, with Italian stage and lighting structure. However, the performance can be adapted to other environments according to the requirement of the curatorship. In terms of equipment, we need:

- 1 concert piano, positioned according to the stage map.
- 4 AC 110v power points, positioned according to the stage map.
- Controlled lighting structure for acting space - spot lights for each musician / musician and a central spotlight on the stage (non-exclusive).
- 1 projector and projection screen.

### 7. Teaser link (for evaluation of the submission)

In this link (<https://www.youtube.com/watch?v=ZJbud73u2GM>) the first version of the *Suíte [en]Quadrada*. The current version, submitted to SBCM 2017, has updates.

### 8. Stage plot and input list

No instrument is amplified via P.A. Only the live electronics performer captures the image of the stage by a webcam, processes the image and projects on a screen at the bottom of the stage. Through the images also generates synthesized sounds that are sent to the central mixer and the P.A.

# The Maxwell Demon

Luzilei Aliel <sup>1</sup>

<sup>1</sup> First ECA – Escola de Comunicação e Artes da Universidade de São Paulo  
Av. Prof. Luciano Gualberto, 158, tv. 3 – 05508-900 São Paulo, SP

first\_luzaliel@usp.br

## Abstract

The Maxwell Demon (TMD) is a comprovisation based on the James Clerk Maxwell experiment in 1871. This comprovisation (composition + improvisation) aims to enable multiple agents to exchange sound interactions through common equipment but not commonly used for sound and artistic practices: the mobile phone. TMD proposes the possibility of technical equality between musicians and non-musicians and sound discovery through interaction and interactivity relations between agents x agents, agent x environment and agents x algorithms. One of the performances of TMD can be seen in: <https://www.youtube.com/watch?v=6sYOe1q7INE&t=138s>

## 1. The Maxwell's Demon (TMD)

The Maxwell's Demon (TMD) is a comprovisation inspired by James Clerk Maxwell's 1871 experiment. In this experiment, the Maxwell Demon is an imaginary creature designed to contradict the second law of thermodynamics, the tendency of every system towards entropy. Maxwell's experiment can be represented as a box with a divider placed in the middle, separating it two compartments, left and right. This partition has a door that can be opened and closed by an imaginary being, called Maxwell's Demon. The demon opens the door to allow only the fastest molecules to flow to one side of the chamber. Only the slower molecules flow to the

other side, gradually causing one side to warm up, while the other remains cool. Thus entropy is reduced. We use TMD as an artistic metaphor focused on sound (rather than thermodynamics) to simulate an imaginary being - in our case, a stochastic algorithm - that seeks to control the sonic outcome to increase or reduce its entropy. Conceptually, we treat stochastic algorithms as *Gelassenheit* entities [Heidegger, 1966; Koutsomichalis, 2011]. A *Gelassenheit* entity has an "independence" in time and space, its dynamics are established by stochastic processes.

## 2. Materials and Procedures

*Materials/Equipments:*

Mobile: We designed a Pure Data (PD) patch running on the MobMuPlat application capable of producing sounds: 1) easy to manipulate; 2) accessible to all agents (through deployment on mobile platforms based on Android and IOS systems).

The mobile screen patch features four rectangles that act as controllers of additive synthesis oscillators. Simultaneous control of up to four banks of oscillators is possible. Four FM synthesis oscillators feature control parameters for frequency, duration and delay. Processes are controlled by tapping the phone's screen and can be turned on and off at any time. The frequencies vary from 220 Hertz to 1320 Hertz. Frequency increments are associated with gestures from left to right.

Beside each rectangle there are three oscillators switches (on/off buttons) and two envelope controllers. One with a condition of attack, decay, sustain and release short and another with envelopes in longer conditions. The left button at the top of the screen controls a stochastic algorithm connected to all the oscillators frequencies. This triggers random changes in each oscillator. The remaining three buttons control delay processing of the sound material. From left to right, there are fixed delay rates of 150 ms., 300 ms. and 750 ms.

*Stochastic Computational Algorithm:* The computer running the automated algorithm is connected to two loudspeakers. The loudspeakers were placed at the two corners of the location where the performance/experiment took place. The agents moved around the perimeter of the environment. We developed a patch (PD) that runs on a desktop computer with similar sonic features as the algorithms for the mobiles. Rather than being controlled by the participants, stochastic automated processes determine when and how sound events will occur. With the touch of the green start button, the entire performance/sound experiment occurs in an automated way. At the ringing of the bell begins the artistic narrative finalizes.

*Sonic materials* - We use an emulated bell (based on FM) that plays at the beginning of the performance/experiment and ends it when it is heard again. This sound is triggered by a PD patch. The entire performance takes five minutes. All sound content that occurs between the ringing of the bells are contingencies resulting from interactions and sound discovery.

*Location and participants* - TMD is designed to run in a small place. Preferably a place with poor lighting. We suggest a hallway, or a small room.

Having Maxwell's procedures as inspiration, we use two types of agents, those with traditional knowledge of music and those with little or no knowledge. In other words, no musical training

is required. There are no limits of participants.

All participants must have a mobile (operating system, Android or IOS), with the application MobMuPlat installed and the TMD patch running. We recommend the existence of internet to enable the implementation of agents willing to participate in TMD.

*Estimated Time* :TMD is designed for three runs of approximately five minutes each. We suggest small discussions of five minutes between the performances. Approximate total of 30 minutes adding performance and discussion.

### 3. Material Summary

- 1 AC 110v power;
- 2 Amplified speakers;
- Small or medium room with poor lighting;
- Interested agents with or without musical training;

### 4. Composer/performer Bio

*Luzilei*

*Aliel:*

<http://lattes.cnpq.br/9729924267239229>

### 5. References

- [1] Heidegger, M. *Gelassenheit*: HarperCollins. 1966.
- [2] Iglesia. D. The Mobility is the Message: the Development and Uses of Mob Mu plat. In: *PdCon16*.NYC. 2016. <http://www.danieliglesia.com/mobmuplat/IglesiaMobMuPlatPaper.pdf>
- [3] Koutsomichalis, M Site Specific Live Electronic Music: A Sound Artist's Perspective. *Proceedings of the Electroacoustic Music Studies Conference, Sforzando!*, New York. <[http://www.emsnetwork.org/IMG/pdf\\_EMS11\\_Koutsomichalis.pdf](http://www.emsnetwork.org/IMG/pdf_EMS11_Koutsomichalis.pdf)>. 2011.

## Author Index

- Adilson Neto, 166  
 Alexandre Torres Porres, 41  
 André Lucas Nascimento Gomes, 107  
 André Martins, 176  
 André Rauber Du Bois, 1  
 Antonio D'Amato, 185  
 Augusto Paladino, 158
- Bruno Mesz, 158, 172
- Caleb Luporini, 99
- Damián Keller, 25, 137  
 Daniel Mélo, 86  
 Daniel Penalva, 99  
 Danilo Rosseti, 123, 180  
 Djalma Lúcio, 144
- Eduardo Meneses, 168  
 Emmanouil Benetos, 13  
 Erivan Duarte, 152
- Fabio Ortega, 81  
 Felipe Falcão, 86  
 Felipe Souza Tanios, 63  
 Flávio Luiz Schiavoni, 75, 107  
 Flávio Schiavoni, 162  
 Franklin Magalhães Ribeiro Junior, 170
- Geraldo Rocha Junior, 99  
 Guilherme Feulo do Espirito Santo, 160  
 Guilherme Lunhani, 99, 162
- Helena de Souza Nunes, 13  
 Helena Souza Nunes, 156
- Jônatas Manzolli, 123  
 José Fornari, 154  
 José Henrique Padovani, 19  
 Juan Pégola, 158  
 Julian Arango, 115  
 Juliano Kestenber, 144
- Kyong Mee Choi, 207
- Leandro Souza, 49  
 Luan Luiz Gonçalves, 107  
 Lucas Samaruga, 172  
 Luiz Velho, 144  
 Luzilei Aliel, 137, 208, 210
- Manuel C. Eguía, 93  
 Marcelo Johann, 33  
 Marcelo Pimenta, 33  
 Marcelo Queiroz, 7, 160, 164  
 Marcelo Wanderley, 168  
 Marcos Woitowitz, 156  
 Matías Zabaljáregui, 93  
 Miguel Antar, 208
- Omar Peracha, 178
- Pablo E. Riera, 93  
 Paul Schuette, 186
- Rafael Ramirez, 81  
 Raul Paiva de Oliveira, 55  
 Renato Fabbri, 99  
 Ricardo Fabbri, 99  
 Roberto Piassi Passos Bodo, 75  
 Rodolfo Pedó Pirotti, 33  
 Rodolfo Valente, 184  
 Rodrigo C. Borges, 7  
 Rodrigo Pereira, 166  
 Rodrigo Ribeiro, 1  
 Rodrigo Schramm, 13, 156
- Rogério Costa, 137
- Sérgio Freire, 49  
 Sergio Giraldo, 81  
 Sever Tipei, 131, 182  
 Shayenne Moura, 164  
 Stephen Sinclair, 67
- Tarcisio da Rocha, 170  
 Thiago Felipe de Miranda Arcanjo, 170  
 Tiago Fernandes Tavares, 55, 63  
 Tiago Tavares, 152
- Vilson Vieira da Silva Junior, 99  
 Vitor Rolla, 144
- Willian Teixeira, 123, 180
- Yonara Dantas, 208



## Institution Index

Caldas University, 115  
Centro Universitário do Estado do Pará, 166  
Conservatorio di Musica di Avellino, 185

Escola Superior de Artes Célia Helena, 208

Freelancer, 99

IFT/UNESP, 99  
Independent, 178  
Inria Chile, 67  
Instituto Federal do Maranhão, 170  
Instituto Nacional de Matemática Pura e Aplicada, 144  
Instituto Paulo Freire, 99

McGill University, 168

Queen Mary University of London, 13

Roosevelt University, 207

Universidad Nacional de Quilmes, 93, 172  
Universidad Nacional de Tres de Febrero, 158, 172  
Universidade do Estado de Santa Catarina, 99  
Universidade do Estado do Rio de Janeiro, 99  
Universidade Federal de Campina Grande, 86  
Universidade Federal de Juiz de Fora, 99, 162  
Universidade Federal de Minas Gerais, 49  
Universidade Federal de Ouro Preto, 1  
Universidade Federal de Pelotas, 1  
Universidade Federal de São João Del Rei, 75, 107, 162  
Universidade Federal de Sergipe, 170  
Universidade Federal do Acre, 25, 137  
Universidade Federal do Mato Grosso do Sul, 123, 180  
Universidade Federal do Pará, 166  
Universidade Federal do Rio de Janeiro, 144  
Universidade Federal do Rio Grande do Sul, 13, 33, 156  
Universitat Pompeu Fabra, 81  
University of Buenos Aires, 93  
University of Campinas, 19, 55, 63, 123, 152, 154, 180  
University of Illinois, 131, 182  
University of São Paulo, 7, 41, 75, 99, 137, 160, 164, 176, 184, 208, 210  
University of the Arts, 186

