Personal insights on the development of a hyper-instrument:

Interactive composition and improvisation with viola and live electronics

Tiago Morais Morgado

Master's Thesis

Institute of Sonology

May 2013

“(...) There are other reasons to use computers in music, however, that have everything to do with the nature of the music performed. (...)”

Robert Rowe in Music Musicianship

Abstract

This dissertation is concerned with the development of a Hyper-Instrument. The motivation for the research was the achievement of mediation between compositional and improvisatory practices with a focus on work with my instrument. This thesis describes the development of such ideas; it is divided into a contextualization and a brief description of the research undertaken.

Acknowledgments

I would like to express my deep and honest gratitude towards all the great people surrounding me during my entire life and all the opportunities to grow as a Musician and Human Being, disregarding the fact that at some points I could have developed more. I have purposely avoided mentioning specific names to not exclude anybody.

Table of Contents

1. Introduction

I'm talking about establishing a balance between formal structure and the physicality of the person performing (...) The balance between structure and physicality is the most intriguing one I can imagine because one attempts to weigh out two highly contradictory but crucial entities. The composer who can handle these extremities is bound to create a lively piece of music. (Waisvisz, 1990, p. 29)

This thesis is the result of a research process that appeared in the sequence of my previous work. My interaction with my instrument and computer was problematic, in what concerns to the need to focus on tapping buttons to control parameterization, once I needed much expressiveness, while keeping the flow of playing, and to focus on what I was doing musically.

I needed to avoid the typical problem of standing behind a mixer in a performance situation, letting the audience close their eyes and listen to the music in a dark room, with no eye contact from the performer, and no relationship between gestural control and sound generation. Solving those issues and creating a system that could, at the same time, crystallize my music in aesthetic terms to allow for new possibilities within it, was a motivation for the development of this research.

2. Hyper Instruments

2.1 Definition of the term

The basic concept of a hyperinstrument is to take musical performance data in some form, to process it through a series of computer programs, and to generate a musical result. Usually this chain of events is meant to take place in real time; however, there is no reason why the same instrument could not be used for composing, experimental, or playing with music structures and sounds in non-real time.

(Machover, 1992, p. 4)

An **augmented or extended instrument adds external behavior to a preexisting one**. A clear example of that is the prepared piano of John Cage, which adds objects to modify the timbral properties of the instrument, adding new possibilities to it.

Another example is the electric guitar, which is not an instrument, at least until it is connected to an amplifier. This is the most pure approach to such an instrument, being the approach that musicians such as Derek Bailey followed. Guitar players usually add to this small combination of devices things like guitar pedals, to prepare the instrument, such as in the case of Fred Frith, Keith Rowe, Hans Tammen, Otomo Yoshihide, among others.

Other example might be the case of people who use sensors to extract information from an instrument. We can situate this on the augmented violin of Max Mathews, on Rose HyperBow, Mari Kimura Augmented Violin, the Hyper-Violin and HyperCello from MIT or Cleo Palacio Quintin Hyper-Flute.

There are two topologies to establish mappings in Live Electronics:

- **Reactive mappings**
- **Interactive mappings**

In the first case, the instruments follow a one to one relationship in the sense of mappings, becoming easy to have predictable behavior, where gestures can easily be repeated. Another topology is interactive instruments. **Interactive instruments contain specific behavior that is partially controlled, such as a parameter being filtered by a certain sort of data generator**. This is can be seen in the early days of electronic music within the works of Chadabe or Martirano.

The term hyper-instrument relates to interactive instruments designed or adapted to use with electronic sensors, whose output controls the computerized generation or transformation of the sound.

Hyper-composition (or meta-composition) systems apply these principles to automatic music generation. Cases of that can be, for instance, something like George Lewis' Voyager, or a system like Nicholas Collins' Autocousmatic.

2.2 A brief introduction to the history of Interactive Music

The following chapter is based on a critical summary of Arne Eigenfeldt's paper *Real-time Composition or Computer Improvisation? (...)* (Arne Eigenfeldt 2007). Music is a field where technique (as applied technology, a consequence of the industrialization process) and scientific knowledge are applied. All these issues tend to collide with aesthetic concerns, of different orders (political, sociological, philosophical, etc.).

With the Industrial Revolution leading to the advent of electricity at the end of the 19th Century we can find a series of new instruments, such as the Theremin, Ondes Martenot, Telharmonium, or the voltage-controlled synthesizer (such as RCA Mark I and II). These were often treated as extensions of the current practice, such as in the case of Clara Rockmore's interpretations of the music of Camille Saint-Saens and other romantic composers.

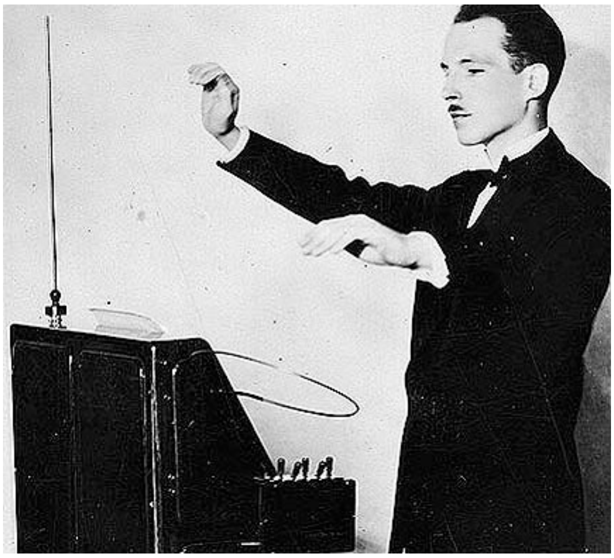


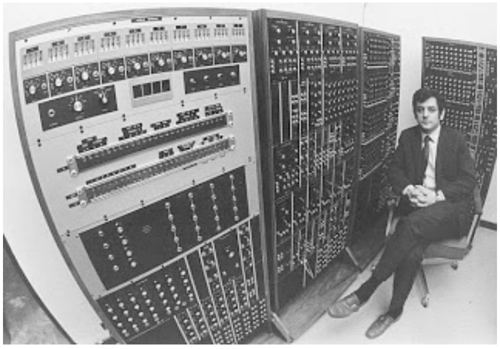
Fig. 1 - Leo Theremin playing the Theremin

The way these instruments were used in such a context was inferior regarding their possibilities, falsely compelling them to compete with the timbral and spectral dimensions of traditional acoustic instruments. However the way those instruments were played influenced later practices in the field of live electronic music. An example where we see these instruments appearing in a different way is the transcription of Ligeti's *Lux Aeterna* for 16 theremins.

John Cage explored indeterminacy in his compositional process, deploying chance procedures to initiate a process by which pieces could take-shape autonomously. This led to a glut of process music, which filtered into electroacoustic music, including some studio-composed works such 'as Cage's William's' Mix (nineteen fifty-two, nineteen fifty-three) and Fontana Mix (1958).

In 1971, Joel Chadabe and Robert Moog designed a system called CEMS. This was the first instance of what Chadabe called interactive composition, *a mutual relationship between performer and instrument*. The instrument's output was not in a direct one-to-one relationship to its controllers. Such a relationship is the basis of *interactive composition*, which Chadabe defines as **a process in which the instrument is set to have an unpredictable behavior that could partially be controlled** (Chadabe 2007).

At the same time Chadabe was to develop the CEMS, Sal Martirano built his own SalMar Construction, a similar instrument controlled by discrete component digital circuitry and performed by touching a large number of touch-sensitive switches. Martirano described his role in performance as follows: “I enabled paths. Better, I steered. It was like driving a bus”



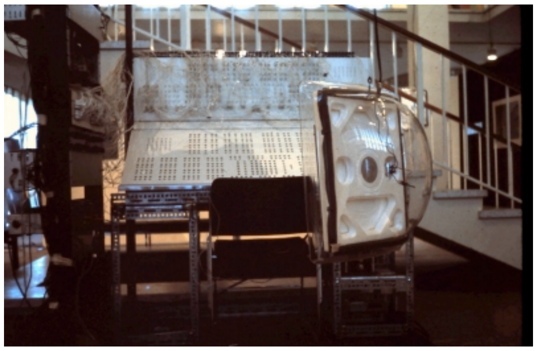


Fig. 2 & 3 - Chadabe and CEMS. Sal Martirano’s Salmar Construction

David Behrman also explored the use of live electronics during 1970s. He incorporated available means of pitch tracking technology in his work, allowing for the control of the oscillator frequencies of a handmade analog synthesizer. In his music he used equal tempered, tonal harmonies, that resulted in *less radical* aesthetics than Chadabe or Martirano.

In these three cases, composers endorsed competing approaches to exploring their musical aesthetics, building interactive instruments that allowed for the creation of gesturally complex music.

In 1977, Chadabe started to work with a system based on the *Synclavier* made by New England Digital. His piece *Solo*, from 1979, used it for the first time. The computer *improvised* a melody in eight voices. Two Theremins were used to control timbre and tempo. By "guiding the system by responsively to control certain variables", the complexity of interaction increased, giving more responsibility to the software instrument.

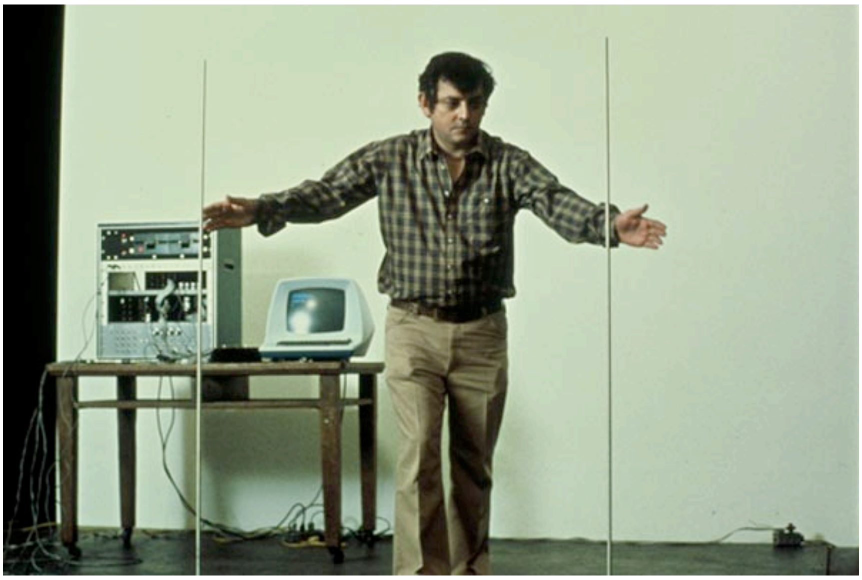


Fig. 4 - Chadabe Solo

By the beginning of 1980s, live electronic music passed through two generations of technology, namely analogue and digital. New technology allowed for the expansion of ideas, yet the predominant aesthetic that favored the unknown within performance over exactitude remained intact. With the appearance of MIDI and the availability of commercial synthesizers, and personal computer in the mid 1980s, there was a whole new spectrum of possibilities to create software and instruments.

Improvisation and interactive computer music are often connected, to a certain extent. In 1977, Chadabe stated the following:

> *Interactive composition is different from typical instrumental improvisation in that an improviser specifies and performs data in reaction to another improviser, but here the composer specifies and performs control strategies rather than data...The difference between instrumental improvisation and control strategy interaction is, then, primarily that of one's position in a control hierarchy, but it is true that in both cases the performer must act and react in realtime*. (Chadabe, 1977)

George Lewis describes his Voyager system as an *automatic composing program that generates complex responses to the improvising musician's playing*. He distinguishes *improvised music*, from music that merely *incorporates* improvisation; furthermore, the notion of improvisation as *real time composition* is implicitly disavowed.

Unlike other systems, *Voyager* acts independently from the control of a human. *Voyager* does, however, clearly resemble traditional acoustic instruments. This is made obvious in the limited information the system extracts from the human performer: pitch and velocity. This data is used to identify what the performer is doing.

Chadabe suggests that *The important quality of random-number generators is what I have come to call complexity. By complexity, I mean that the rules which determine a certain succession of events are underlying and elusive and not evident in the succession of events they generate*. (Chadabe, 1983)

In a Similar way, Lewis uses randomness to control “melody, harmony, orchestration, ornamentation, pacing, transposition, rhythm, and internal behavior decisions.”

> *The use of white noise allows me to filter the resulting stream of numbers in a conceptually simple and orthogonal fashion, where "tuning" the program for a desired musical result, or setting up input response parameters for interaction with improvisers, ultimately becomes a process of setting percentages.* (Lewis, 1999)

By the end of the nineties, affordable hardware became fast enough to allow for live signal generation and processing. Many of the same signal processing routines that were formerly restricted to the studio could now be used in real time. The result has been a dramatic change in live electroacoustic music: it became based upon the development of timbral gestures.

Chadabe’s piece *Many Times(...)* uses just such a strategy: recording a performer, transforming these recordings, and playing them back in a multichannel sound system. Chadabe claims that the *transformations, generated by the software that animates the instrument, are essentially unpredictable. Because of their unpredictability, they provide the performer with something to react to*.”

Guy Garnett, in his 2001 paper, describes new ways to make a work interactive, including extending a traditional instrument through processing. Rowe (1993) also reviews a wide range of interactive systems including his own, Cypher. This system consists of two major real-time components, the ’listener’ (analysis) and the ’player’ (composition). Each component consists of interrelated agents, which constitute agencies (Minsky 1988).

2.3 Hyper-Instruments, case studies

2.3.1 A critical Summary of Daniel Trueman reinventing the violin (1999) “the infinite violin: the descontructed violin reconstructed”

In the third chapter of his book *reinventing the violin* (1999) “*the infinite violin: the descontructed violin reconstructed,”* Daniel Trueman talks about his research on the topic of interactive violin, describing also similar or different approaches by other people related to this topic.

Trueman mentions that the developers of the hyper cello at MIT focused on augmenting the instrument without modifying it: *We sought to develop techniques that would allow the performer's normal playing technique and interpretive skills to shape and control computer extensions to the instrument* (Paradiso, 174). My own approach also is quite similar to this.

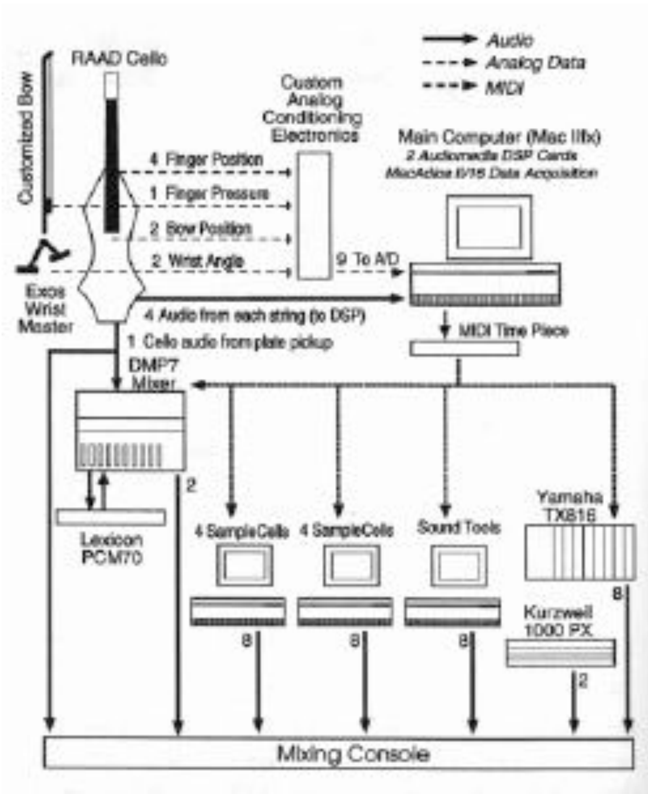


Fig. 5 - Tod Machove’s HyperCello Hardware Diagram

In a different way, Jon Rose, with whom I clearly identify myself (but not so much with interaction design), points out that:

I've tried to keep a one to one connection between violin and digital instrument by using only one midi channel in performance. Sort of basic monophonic solidarity! For me, an important aspect of expression comes out of pushing the natural physical limitations of an instrument to the edge of its possibilities, this includes digital ones as well. (http://www.euronet.nl/users/jrviolin/chaotic.html, quoted from Trueman, 1999). **I am against gestures motivated by the design of an instrument**; that's one of the reasons why I try to conceive, in an adaptive way, the design of an instrument to the specificities of my playing.

Fig. 6 - john Rose HyperString Project

According to Trueman, *My own R-Bow (designed and built with Perry Cook) is similar to Chafe's bow, but rather than a bend sensor, we mounted two force-sensing-resistors (FSRs) between the stick and hair on light foam (*Trueman, 1999*)*. I do not like this approach once it completely changes the whole dynamic of playing. I don’t think it makes sense to sacrifice the playability of an instrument for an increased sensory aspect.

Fig. 7 - Daniel Trueman’s Rbow

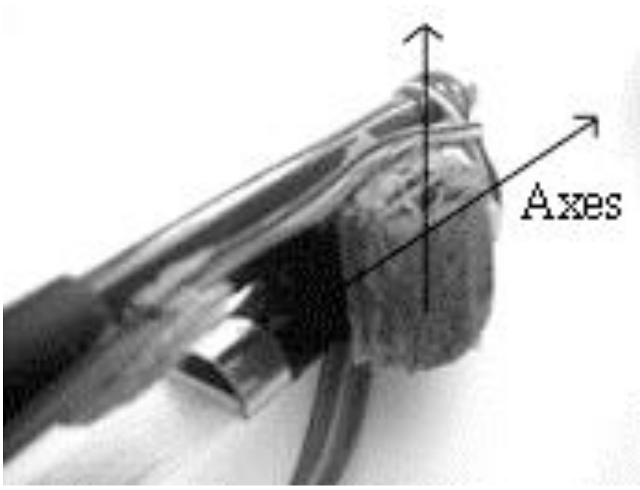


Fig. 8 - Daniel Trueman’s Rbow Accelerometer Placement

2.3.2 Frances-Marie Uitti Augmented-Cello

In the paper Augmenting The Cello (Freed and Uitti, 2006), Adrian Freed and Frances Marie Uitti describe software and hardware enhancements on an electric 6-string cello. The starting point for the project was a 6-string cello built by Eric Jansen. A new solution to the problem of changing tunings of the open strings; a matrix of switches and pressure sensors installed on the instrument, a novel bowed rotary encoder; and the software used, are all described.

Uitti uses many nontraditional tunings allowed by multiple stops and two bows. Such an issue becomes problematic because of the physical specificities of the instrument. One approach to support different tunings is to use independent pitch shifting DSP algorithms on the signals captured by piezoelectric pickups under each string at the bridge. In a previous project on hex guitar signal processing, the CNMAT team identified several problems with pitch shifting:

1) Numerous noticeable artifacts in the shifted sound.
2) Conflict between the acoustic sound and electronic sound in live performance
3) Unacceptably long latencies especially for low-pitched strings.

Fig. 10 - Frances Marie uitti CNMAT Cello

CNMAT Team’s solution to these problems was to augment the cello by adding a mechanical tension-modulating device at the heel of the instrument. This device was originally developed for guitars by Hipshot Inc. [10]. Foot control is commonly used in live performance especially with computer-based scores. However it is a solution that brings a bowed string performer out of physical balance, as stated in the paper.

For the stopping hand CNMAT Team provided a row of Force Sensing Resistors. On the other edge of the neck the team installed a continuous pressure-sensing strip accessed typically with the thumb. The team also installed a switch array directly below the bridge and an arrow of circular FSR’s at the top of the body of the instrument.

The CNMAT Team attempted to sense the string’s stopped position using a resistive strip designed as a “ribbon” controller, but found it too wide and short for this application. For the bowing hand the CNMAT team introduced a novel application of a rotary absolute position encoder.

These analog signals are conditioned, converted into digital signals, serialized and aggregated into an Ethernet stream, which was then processed by custom software in Max/MSP. The CNMAT team elaborated and augmented ideas originally developed for an earlier polyphonic guitar project [15] to reflect Uitti’s aesthetic needs.

One programming challenge is to give the performer as much meaningful control as possible without overwhelming them with parameters that they will find useless or, worse still, distracting. To that end, overall control of the performance sub-patches was managed using a combination of OSC (Open Sound Control)[14] and the pattr family of objects. Each of the hardware sensors was given a unique address in an OSC namespace, allowing individual sub-patches to tap into the appropriate control data.

The CNMAT team used a separate bank of five resonant formant filters for each string. I am not specifically fond of using resonators with acoustic instruments. This is to do with the fact that I do not find the sounds particularly interesting. The key idea of the double stop convolution patch was to use a separate convolution for all the double stop combinations and to process and spatialize the output of the convolved pairs independently.

Convolution works well in this situation because sound is only output if there is a signal in both inputs of the convolution. Two patches were combined in the panning granulator effect, with the intention of surrounding the direct sound of the cello with a diffused aura of related fragments. The panning patch diffused the sound in a circular array, maintaining a 180-degree separation between each channel of each granulator.

2.3.3. Cleo Palacio Quintin HyperFlute

In her paper interactive composition and improvisation on The Hyper Flute, Cleo Palacio-Quintin describes the project that she has been developing since 1999. The extended flute is interfaced to a computer through electronic sensors, enabling the control of various digital sound processing parameters.

Quintin's interactive composition strategies for the hyper-flute have been influenced largely by her practice of improvised music. The original design of the hyper-flute was done using a MicroLab, originally designed by Scherpenisse and Broek at the Institute of Sonology of The Hague. It offers 32 analog ports, a matrix of 16 ASCII keys, and an ultrasound measurer. Inspired by the design of the meta-trumpet.

The hyper-flute design was mostly based on sensor placement. From my perspective, an instrument like this would benefit into a much larger extent from having a complex sound feature extraction system, rather than having a huge amount of sensors placed in itself. This requires wires, which affects the performance of the flute by changing the weight of the instrument.

Quintin later states that *using an interactive computer system linked to an augmented instrument, the performer has to develop a relationship with different types of electroacoustic sound objects and musical structures. These relationships correspond to the fundamentals of musical interaction. The computer part can be supportive, accompanying, antagonistic, alienated, contrasting, responsorial, developmental, extended, etc* (Quintin, 2008).

All the musical structures included in a mixed piece have different roles. Some affect the microstructure of a musical performance, others affect the macrostructure of the piece, and many are situated somewhere in between. The interaction between the performer and musical structures vary. The structures can also have different levels of interactivity between themselves. We could divide them in 3 basic distinct types:

• The original acoustic sound is modified by live processing, controlled through the gestural interface. For example, the computer is modifying and/or extending the acoustic sound itself, by routing it through filters, harmonizers or delays. The computer is used as a direct extension of the performer's acoustic instrument.

• Sound is synthesized in real-time using the various interfaces (gesture information and sound analysis) to control different parameters. Synthesis can respond to the performer's gestures without being directly linked to the acoustic sound of the instrument. Control and sound data can also be recorded and used later during the piece, permitting time stretching and compression.

• An independent soundtrack can accompany the flute or play by itself over the course of the piece. It can be prerecorded, or generated in realtime with the use of computer algorithms. This type of structure is completely independent from the performer's actions.

Fig. 11 - Cleo Palacio Quintin playing the HyperFlute

2.3.4 Nick Collins Autocousmatic

In the paper Automatic Composition of Electroacoustic Art Music Using Machine Learning (Collins, 2012), Nick Collins (Department of Informatics, University of Sussex) *presents Autocousmatic, an algorithmic system that creates electroacoustic art music using machine-listening processes within the design cycle*.l.

A great challenge to the automated production of musical works is the critical role of the human auditory system within the design cycle. Human compositional activity over a musical form provides for continual feedback at multiple timescales, from selecting and refining momentary material, to the control of flow between sections and across the whole work (Roads 1985; Eagle stone et al. 2008).

Algorithmic composition has not extensively engaged with problems in the audition of the generated music, including the psychology of musical form (Collins 2009).

Previous algorithmic composition "critics" (software modules that evaluate the output of compositional algorithms) have operated mainly in a pre-established, symbolic domain—for example, as fitness functions in a genetic algorithm search (Miranda and Biles 2007).

Later on, Collins states that *Although full equivalence of human and machine-listening capabilities remains out of reach at present, much research advancement has occurred in this domain in recent years (Klapuri and Davy 2006), and continues to be pushed particularly through music information retrieval (MIR) research (Casey et al. 2008). Indeed, machine listening applied directly to audio signals has become a strong feature of work in live interactive systems (Rowe 2001; Hsu 2005), and it is somewhat surprising that it has not taken place to the same degree within algorithmic production of fixed works.*

My interest in these approaches is not related to the idea of rendering *Aucousmatic* pieces, but deals with the possibility of efficiently inserting pre-recorded material derived from my instrument in my playing. This allows me to exploit the complexity of the things I am doing musically.

Collins mentions some score generation environments, such as Csound, Karlheinz Essl Replay PLAYer (2000-2007), Leafcutter John's *Forester* series of programs, ThonK, or Peter Traub's *bits & pieces: a sonic installation for the World Wide Web* (1999), which gathers files from internet and generates new mixes.

According to Collins, the system starts from a folder of given input material, then a duration for the piece is specified. After initialized, the program enters a number of routines that are constantly iterated to derive a piece.

Collins mentions that *In Autocousmatic, sections are created with different guiding parameters, including such aspects as the overall density of events, and the abruptness of transitions into and out of the section. Each section is assigned one to four source files as its primary materials. When two or more source files are available in a given section, there is the potential for cross-synthesis of materials in the processing stage.*

About the processing stage, Collins states that interesting parts of the files are selected, and recursive sound transformations take place. After being processed the file is tested to determine whether or not it is suitable to keep being processed or for taking part of the compositional material for the given section, as seen in pioneering algorithmic composition by people like Hiller and Isaacson.

According to Collins, sound analysis is based on using percussive onset detection and perceptual loudness, as well as on RMS and absolute-peak amplitude.

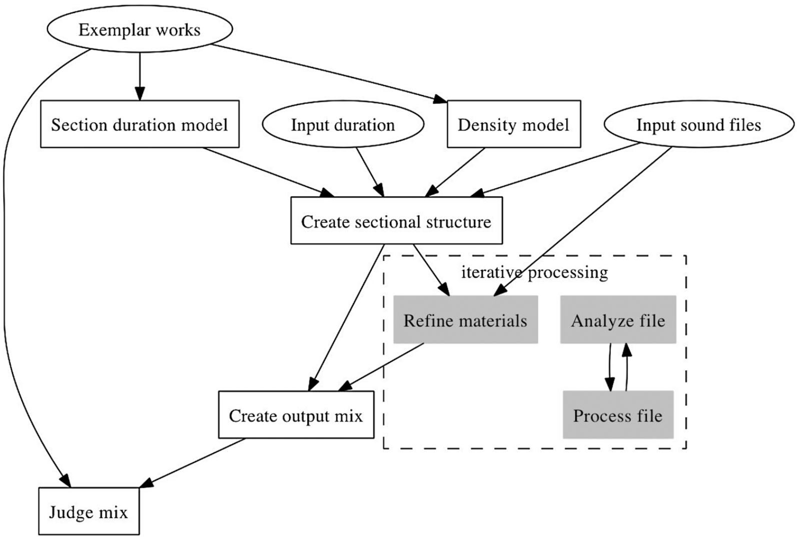


Fig. 13 - Nick Collins *Autocousmatic* software design diagram

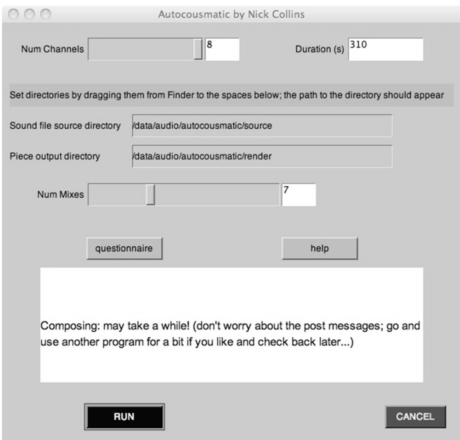


Fig. 14 - Nick Collins Autocousmatic Gui

2.3.6 George Lewis Voyager

In the paper *Too Many Notes: Computer, Complexity and Culture in Voyager* (Lewis, 2000), George Lewis discusses his computer music composition, Voyager, which employs a computer-driven, interactive "*virtual improvising orchestra*". This analyzes an improviser's performance in real time, generating both complex responses to the musician's playing and independent behavior arising from the program's own internal processes.

According to the author, *In Voyager, improvisers engage in dialogue with a computer-driven, interactive "virtual improvising orchestra." A computer program analyzes aspects of a human improviser's performance in real time, using that analysis to guide an automatic composition (or, if you will, improvisation) program that generates both complex responses to the musician's playing and independent behavior that arises from its own internal processes.*

The programming of the whole environment obeys to all these subsets of different behaviors, clearly typical of human interaction in a context of free improvisation. Besides this, the software tries to embed compositional knowledge, from an algorithmic composition perspective, of Afro-American free improvised music. When I think about conceiving an interactive system, I give value to the possibilities of creating and representing structures of music knowledge capable of creating never before heard music, by allowing sample manipulation of recordings of my playing.

3. The development of my Hyper-Instrument

Nevertheless, computer systems cannot function in improvisation unless they are programmed with ways to make some sense of the context around them and to react in accordance with that sense. This brings one to the topic that we refer to as machine musicianship, i.e. the realization of concepts concerning human musicianship on a computer program, coupled with emulations of human cognitive processes (Rowe 2001).

The motivation for the development of my Hyper-Instrument was, as stated previously, solving problems in the domain of physicality. At the same time, merging all the dimensions of my electronic composition work with viola performance through interactive music.

It is very clear to me through analysis of the way I develop my lexicon within free improvisation; that the work within acoustical improvisation, and with live electronics ends up influencing each other. The same is applicable to collective improvisation.

When extending instruments with electronics, we theoretically have an unlimited amount of resources. In the sense of the number of possibilities that we can use for mapping, sound processing, sampling, etc., contrary to what happens with a physical or acoustical instrument, in which the instrument sets limits to what may or may not be done musically.

It's good to base our practice on a limited amount of resources that we can easily master in the sense of a learning curve. Mastering a live electronics setup is as demanding as mastering an instrument. It is necessary to spend a large amount of time to truly take advantage of its potential. It is a good thing to keep developing an instrument, as it is mastered from a practice perspective.

That is the advantage of being a developer, a composer, and a performer at the same time. Choosing and/or building the set of tools that we are going to use as our live electronics setup implies the same sort of compositional thinking that we apply to the composition of a piece. In the development of my instrument, I had to first think of what kind of prominent gestures and playing style I wanted to use within my instrument, to then define a set of tools.

The whole Hyper-Instrument was a process of research during three years. Three years in which I developed a software component by myself, and an interactive bow with the help of Lex van den Broek. It changed several times over the course of these two or three years, and I still have ideas to keep working on.

It would not have been possible without the help of people like Richard Barrett, Joel Ryan, Johan van Kreij, Frank Balde, and Pal Berg; with their infinite wisdom and generosity as some of the best teachers that I have ever had.

I was also assisted by people in Portugal including my teachers Rui Dias, Gustavo Costa Filipe Valpereiro. In addition the inspiring and generous talks with Carlos Zingaro in the early stage of the project, about his work within live electronics and his experience playing with Richard Teitelbaum, in his cyberband and in the interactive opera Golem.

Finally most of my friends and colleagues who I played with, that gave me constant feedback and commented on my playing, the way as the instrument was sounding, advice on how the software could develop, and on the way my music could develop.

I had the opportunity of testing the Hyper-Instrument several times, namely solo and collaborative sessions and gigs with Raoul van der Weide, Leo Svirsky, Chris Iemulo, and Katt Hernandez; with V4W.Enko, with Nicola Casetta, with Andreas Lo-an-Joe and with the Sonology Electroacoustic Ensemble.

3.1 General structure of a Hyper Instrument

Most of the hyper-instruments based on acoustic instruments have a common sketch traced by the existence of both a controller, and a sound interface as an interface. For computing they are usually split in three parts, the first being the analysis framework: the second the whole mapping environment, and finally the sound manipulation component, usually compound by synthesis, processing, and sampling algorithms. Finally there is usually a transducer interface or PA System and an Interface (monitoring) for output.

Examples of controllers can be the Thunder (Donald Buchla), The Web (STEIM), Radio Theremin (Robert Moog), Radio Drum (Boie, Mathew and Schloss), Lightning (Donald Buchla), Body Sensors (flex sensors, accelerometers), ultrasonic tracking systems, and video tracking systems. The controllers have as a specification the fact that they can be both directly mapped or filtered by analysis at the same time.

The most common procedures for analysis range from sensors to FFT and DSP based computation, to all sorts of logical and statistical means of data representation and analysis. Then in between we have sound processing procedures of different orders. Finally output, a PA and Monitoring.

The system had over five versions during the academic year 2011-2012, and two versions during the academic year 2012-2013. In between I started to define my hyper-composition system, and ended by defining a completely different path for dealing with the issue of interactive composition.

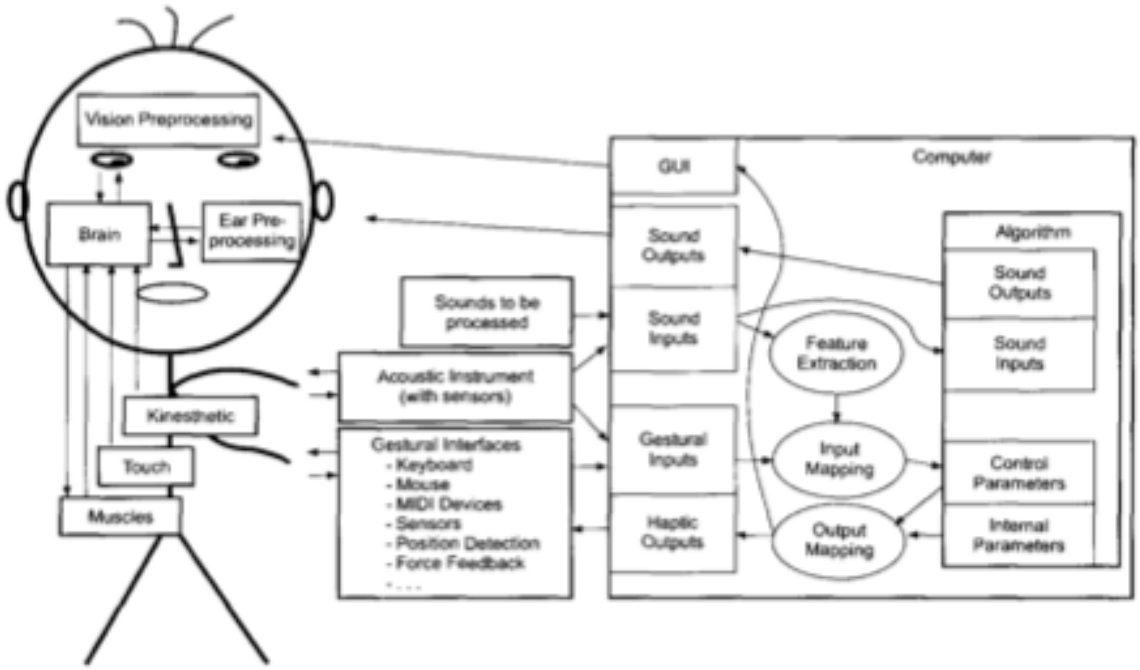


Fig. 18 Diagram of a Regular Hyper-instrument

3.2 The Several versions of the system

In the first version, dating from October-November 2011, the whole feature extraction system was implemented in MaxMSP. It already met many of the parameters that I implemented later on in the final version. There was a mediative patch in MaxMSP for Mappings and the whole sound processing was being hosted in Ableton, comprising some of third party MaxPatches and plugins. This was similar to the setup that I used several months earlier to play in a workshop with Fred Frith and Mark Dresser.

I think that the fact that I had to work with a precise deadline for having the system ready to play with Peter Evans was a big stimulus. The biggest disadvantage of the system was its degree of instability and heavy CPU load that ended up leading to an unavoidable crash of the system, preventing me from playing properly with the electronics during the concert. However I think it was quite successful, advancing the project in this early stage.

In the second version the feature extraction system was implemented in MaxMSP. The mappings were all done inside Max. For Sound Processing I was using some of my Max patches, as well as third party MaxForLive Devices. For sons spatialization I was using a Max Patch coded by myself, which allowed 4 and 8 channels sons diffusion. There was an effective port to MaxMSP. The system was no longer running on Live, proving to be quite much more stable.

At the same time, the system was, effectively running in a much more sober way, with a largely improved GUI. There were two many concerns relating this version of the system. Not everything in the GUI was very accessible, tending to be complicated when accessing all the menus and all the sections of the patch.

Then I also felt like the system was not working in such a meaningful way regarding DSP (and when I refer to DSP, I mean, effectively sound processing). Finally I felt from my perspective that the fact that I switched to Max allowed me to have something more fixed, close to a standalone application, but at the same time more difficult to manage, to update and to develop.

In the third version there was an attempt to commit to a new way of expressing my ideas on hyper-composition, mostly by using third party patches, in which I could embed some very simple algorithmic knowledge to drive synthesis parameters. The GUI was updated. There was an update on the sound spatialization modules that I was building, as well as on the sound processing modules. But there was not a very refined sound processing treatment. Besides this, there was not a clear relationship between input and output

The forth version met an updated GUI. The Hyper-Bow met its first version. Built with Ipson Lab Compact, using an accelerometer, a pressure sensor and one button. The signals arrived into Max using a patch coded by Broek and Krei. I think that the fact that the GUI was updated made everything more accessible in the patch. The bow and the GUI were the two biggest advancements.

With the bow I think that it was quite successful in the sense that I was able of achieving something that was close to the final version of the bow developed one year later, by using more cost efficient technology. However, it is true that although right now I am much more close to finding an effective use of the overall parametrization and possibilities of the system, both the sound and bow feature extraction were being misused.

With the fifth version, dating from the end of the 2011-2012 academic year there was an implementation of SoundHack Pvoc filters, including the +Pvocloop, +SpiralStretch, +Phasemash, and +PitchShift. There were also prates in the GUI. Mappings were done with junXion. In this fifth version there was, effectively a much bigger relationship between the input sound of the viola and the out-coming processed sound. Once only viola sounds were being given as an input. The GUI met its most successful of all MaxMSP versions. Overall the system managed to be the greatest advancement of the whole year. I was happy with it.

The main disadvantage of the system were the mappings which didn't match my wish to have a system that could adapt to any playing style. I also needed higher quality sound processing with fewer artifacts. The system proved to be quite successful at enhancing density but not so much in contributing to enrich the sound qualities of the viola, by allowing sounds that could not be produced by any other means.

In the sixth version the environment was re-implemented in MaxForlive, Ableton and junXion. There was a possibility for multichannel sound; no performances were made with it. I committed to the first successful attempt to merge all the dimensions of my work through an interactive patch in MaxMSP, coded by me. The principles in the patch worked in a more effective way with pre-recorded viola sounds rather than with complementary sounds of expired genres of electronic music, once the medium (viola and electronics) remains.

Instead of using Ipson Lab compact, I ended up using the sensors of a Wii-remote attached to my bow. I had an accelerometer and two buttons; it ended up by being much cheaper to produce such a bow. About the implementation itself, it proved to be stable although effectively very heavy with CPU load. A big advantage was that it was more flexible, allowing me to try out something at any point, without having to loose one month reconfiguring the whole system.

In the seventh I encapsulated my feature extraction system back in MaxMSP. By doing so I used a smaller amount of CPU, than in Ableton Live and MaxForLive. One problem of it was that as all the dynamics-enhancement-processing for the envelope follower were off, it ended up being less efficient in onset detection, among other parameters.

In Ableton Live, the patch was loading the CPU in 8-12%, in MaxMSP in 2% to which were added the 8 percent of running MaxMSP without any running patch. All the synthesis, mappings, and sound diffusion ended up happening in SC3. Within less than one month, I was already capable of implementing all my synthesis in SC3, and solving a lot of problems that I was not capable of solving otherwise, because of limits of MaxMSP architecture.

For synthesis I used mostly warp1 ugen, with an example coming from Pal Berg tutorials, that I hacked by myself to create my own trigger methods, and ways of instantiating the patch, with the help of the framework of Matus Kobolka, in which we both worked collaboratively.

For the mappings, I just threw all my feature extraction with more that 25 parameters in SC3, selected the most active parameters and map them to the control of everything using certain interaction strategies, like controlling synthesis for mediating control of parameters, inverting, interpolating, and gating the vales, using sets of conditions, switch, if do while and many other sort of c based scripting approaches.

For sound diffusion I used multichannel expansion techniques. That allowed me to spread the components of the synthesis thru different speakers. For Visuals I used QC with very simple patches, controlled by SC3, sing the same triggers that I used for the synthesis. The bow was not changed.

Hyper-Viola	Version I October/ November 2011	Version II/ anary 2012	Version III feb 2012	Version IV march 2012	Version V may 2012	Version VI an-march 2013	Version VII april/may 2013
Featre extractionhosted in	MaxMSP	MaxMSP	MaxMSP	MaxMSP	MaxMSP	MaxForLive/ Ableton	MaxMSP
Sond Processing hosted in	Ableton	MaxMSP	MaxMSP	MaxMSP	MaxMSP	Ableton	SC3
Mappings done with	Max	Max	Max	Max	nxion	nxion	SC3
Tools for sond processing and sampling	Third Party Max4live Devices (Robert Henke, Plggo4Live) and plgins	some MaxMSP patches of mine, third party MaxMSP patches (vrk, Nabyas Sakodna, etc.)	some MaxMSP patches of mine, third party MaxMSP patches (vrk, Nabyas Sakodna, etc.)	some MaxMSP patches of mine, third party MaxMSP patches (vrk, Nabyas Sakodna, etc.)	SondHack Pvocloop, Pitch shift, phasesh, spiralstretch	SondHack Pvocloop, Pitch shift, phasesh, spiralstretch	SC3 Synthdefs Coded by me
Tools sed for sond spatialization	N/A	A maxpatch coded by myself	A maxpatch coded by myself	A maxpatch coded by myself	A maxpatch coded by myself	Stereo/Robert Henke MaxForlive devices	SC3
Tools sed for visals	N/A	N/A	N/A	N/A	N/A	N/A	Qartz Composer
Hyper-bow							
Platform of Development	N/a	N/A	N/A	Ipson Lab Compact	Ipson Lab Compact	Wiimote+Nnchck	Wiimote+Nnchck
Sensors	N/A	N/A	N/A	Accelerometer, pressre sensor, , one btton	Accelerometer, pressre sensor, one btton, one infrared emitter/receiver	Accelerometer, two bttons	Accelerometer, two bttons
Software sed for collecting sensor data	N/A	N/A	N/A	MaxMSP VKR/ Lex Patch modified by me	MaxMSP VKR/ Lex Patch modified by me	nxion	OSCulator
Hyper-composition							
Tools sed	N/A	N/A	MaxMSP Third party patches	LiSa	LiSa	A MaxMSP patch coded by me	the system becomes the compositional tool. a Max patch created by me is used to compose with these sounds. the compositions are autonomous from the improvisations, but I am aiming for integrating them by controlling sample playback gesturally in sc3

3.3 Feature extraction system

At the current stage of its development my feature extraction system is fully implemented in MaxMSP. It comprises gestural parameters and parameters related to the quantization of musical phenomena. Remembering this, I find it relevant to mention that the system comprises parameters directly derived from sound analysis, through FFT, and from hardware-sensor placement.

The sound component of my feature extraction is derived from parameters such as spectral loudness, brightness and noisiness; and from an envelope follower, that makes a more accurate measurement of pitch and amplitude values rather than the one actually provided by the tristan jehann. I have an integrator, that keeps tracks of onsets and makes an average of the pitch in coming from the space in between each onset, to determine which note is being played.

In the X axis, corresponding to frequency, we create streams of Lists, that are reseted and output according to onset detection. At the output the values end up by being averaged determining the most common element in the list that ends up by being recognized as the current pitch value.

Detecting an attack of a percussive instrument or, into another extent, a violin, or viola, ends up using peak measurement. Amplitude and energy are measured by using an integrator (the same mechanism present in most of DAW Vu meters). In my case, when I measure the onsets if there is a difference of energy around 64.7% than we have a new attack. The resulting information is treated from an arithmatic, statistical and logical, point of view, using simple operations. That results in the many parameters available

The resulting parameters comprise detections of new phrases, or even sections, which play, indeed an extremely important role, on the cueing of new presets or controllers in my SC3 instrument, to things like pitch, rhythm, or spectral parameters, which play a role on identifying what is actually going on musically wise, and mapping the most active parameters, indicators of the predominant gestures in my granular.

The approach with the bow is quite simplistic The sensors end up by connecting to my Mac through a bluetooth connection. The most efficient HID for the bow is the oscillator, which receives the data from the Wii-remote. At the current stage of development, the data is routed to the mappings; it is coupled to the sound analysis to measure the most relevant parameters for mapping at each instance, as well as delivered back to MaxMSP for GUI purposes.

3.4 Parameters

3.4.1 Pitch parameters

Pitch parameters are calculated by the application of arithmetic operations directly on top of the frequency analysis. For pitch registry average the pitch is injected into a stream of values. This is done by using a zl stream object set to a specific range. Than the values are averaged, using the zl median object, and the resulting values are actually scaled to a float range between 0 and 1.

For consonance values are ordered in a descendent order, subtracted. the result values are routed thru a variable into a table, that as classification of 12 tone intervals according to their degree of consonance/dissonance, which are output, routed thru a zl stream and zl median, and scaled using zmap, just like in pitch average.

For registry variance values are set in a descendent order, subtracted, and placed averaged, and scaled the values are passed by a modulo 12 operation. Averaged and scaled.

For melodic interval the same as register variance, but this time with values actually resulting from pitch class measurement which, are the values running on top of a modulo 12 operation.

3.4.2 Rhythm

The rhythmic parameters result of processing the data of onset detection. Arithmetic and logical operations are applied to the values, which are streamed, averaged, and scaled to a range of 0-1, which is the case of density.

For periodicity, if the floating point ratio of the duration of two sequential events is in between 0.25 and 0.75 a 1 is triggered, otherwise a 0 is triggered. A stream of 10 values is made, and the numbers are actually summed and scaled.

For rhythmical regularity, the absolute value results from the subtraction of a median of two values that result from both density and periodicity. The values are scaled and inverted and this results in the parameter

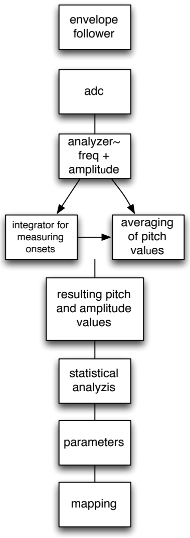
3.4.3 Phrasing and section parameters

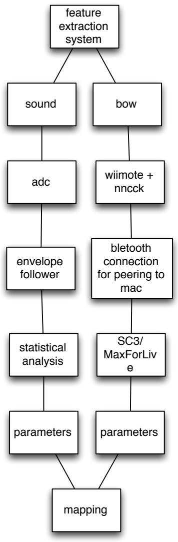
The detection of new phrases is calculated with deviations from the average of duration of time between onsets. All the other phrase parameters, result from applying the rhythmical algorithms to phrases.

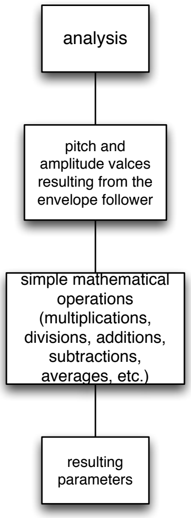
The detection of new sections is calculated with deviations from the average of duration of time between phrases. All the other section parameters, result from applying the rhythmical and phrasing algorithms to sections.

3.4.4 Spectral parameters and bow.

The spectral parameters directly result from the FFT implementations of the analyzer~ object by Tristan ehann. The bow parameters result from the wiimote sc3 class, which is coupled to my MaxMSP feature extraction data as well as to the MaxMSP Graphical user interface.







Pitch	Rhythm	Spectral	Phrasing	Sections	Bow
pitch average	New Attack	Loudness	new phrase	new section	Acceleration X
pitch consonance	Rhythmical Density	Brightness	phrase density	section density	Acceleration Y
register variance	Rhythmical Periodicity	Noisiness	phrase regularity	section regularity	Acceleration Z
pitch class average	Rhythmical Regularity		phrase periodicity	section periodicity	Button 1
melodic interval average					Button 2

Fig. 19 - Table with list of feature extraction parameters of the Hyper-Viola

3.5 Mappings

In the last chapter of the thesis we spent some time discussing the implementation of the feature extraction model for the Hyper-Viola. In this chapter, we will focus on explaining the way as mappings are done in the system, remembering the available parameters list.

The number of available parameters in the feature extraction system is far much larger rather than the total amount of parameters available for mapping. Such an issue might clearly be, contrary to what might initially be thought, a huge advantage, once practically any gesture can be tracked on an efficient way and mapped accordingly, if things are properly done.

So, all the controllers parameters in my SC3 instrument are mapped to two parameters, namely an X and Y parameter. It was quite a challenge to figure a way to take advantage of such a large amount of parameters in the system.

I detect, with all my feature extraction, the two most active parameters within the system. That would theoretically contain the most relevant data to be mapped, and make the mappings. All the synthesis data are mapped to these mostly active parameters.

So, remembering this, I think it's important to briefly discuss the was as, the sc3 instrument is instantiated. At each new phrase there's 30% of possibility of instantiating new copies of the synthdefs.

A number of combination of parameters is selected from the available list of controllers. The parameters are combined, and a new synthdef is instantiated. The minimum number of synthdefs playing at a time is 0, the maximum three. Bearing this in mind, the way as buffers are filled, is that in each new section, the first phrase fills the buffer.

4. Points of Innovation of The system

Since my starting point of work within hyper-instruments that It was never my intention to innovate in the overall way as things are done in this specific field of computer mdic research.

Rather than that I tried to solve problems in the way as I was, into a certain extent approaching live electronics, and from that starting point managing, into a certain extent, a mean of establishing a sense of unity within my work. So trying to put things together and by putting things together, getting a way as unifying my musical lexicon, and coming from that getting a mean of introducing some innovative approaches into this field, from a technological point of view seems to me like a clear possibility.

I think that the unification of my lexicon comes much more from limiting myself to the viola and to the hyper-instrument, let my ways of approaching it expand, and then use a micropoliphony patch that I created to compose tape pieces and installations with the recordings of my material. Eventually keep develop this patch in SC3, and couple it to my instrument

I think that innovating aesthetically wise might come from this aesthetic crystallization. And coming from that point of view, the way of technological development that I plan to introduce in my system, might, effectively, be a wonderful way of expanding this into another dimension of technological realization, and from this point being capable of innovating in this field.

5. Conclusions and Future directions

Although already being a topic of occupation for over 3 years, the Hyper-Instrument is still at a very early stage of development. Mastering this tool, is an issue of practicing, while at the same time keep developing the system, from a software engineering perspective. It is the basis for developing my work as an instrumentalist and improviser, while at the same time composing, and putting together all the dimensions of the work in a single framework.

One of the things that I would, effectively, like to do, will be, into a certain extent, porting all the system to SC3, including the feature extraction system, and develop it all from there, once SC3 has proved, in the single month of regular utilization that I took with it, to be the ideal tool for keep developing my system, once most of the things that I was actually trying to commit to in MaxMSP are much easily implementable in SC3, I can scale into much larger degrees of complexity from an algorithmic sound synthesis perspective, rather than the one that MaxMSP, into a certain extent was allowing me.

The GuI might keep being developed in MaxMSP, but in short mid term, I want to commit to visual representations of the data of the system through simple live visuals, that allow me to bring the GuI into another stage of development.

I also feel that I wold like to add keithmcmillen k-bow to my hardware setup, once no matter how long I dedicate myself to keep developing the interactive bow, it will be impossible to accomplish a bigger degree of complexity than K-Bow, which is a system that comprises over 25 years of research, of some of the most experienced people in the field. Besides the Keith mc mcmillen, I wold like to add the keithmcmillen softep. Even though, by now I am happy with the wiibow, once it works perfectly and it is highly cost efficient.

I also feel that the future of my system, is into a certain extent, dependent of exploring more complex synthesis models than the ones I am sing, by trying to commit to implementations of models such as granular synthesis, phase vocoding, corps based concatenative granular synthesis, spectral mosaicing, and another synthesis models, that I might, in mean time, develop by myself. This can be done either by using Kyma or sc3 with OpenCL, or even both at the same time.

Another thing that I wold like to explore, wold be, into a certain extent, exploring more complex sound processing technique s rather than the ones I am sign right now. The way as I feel I should do such an issue is, into a certain extent, trying to take advantage of nowadays graphical computation power, so that I can within SC3, develop DSP applications for my sound processing taking advantage of the powerful OpenCl

I always have been passionate about video-games, although I have not been playing them for a long time, as well as graphics technology, so I believe, into a certain extent, that developing means for visualizing graphical content by sing games engines, is something that I should explore. During much time Quartz Composer has proved to be the ideal tool for me. Since some months ago, I have started to realize, that the best tool for keep developing my work wold be unreal Development Kit with OSC, sing the CCRMA implementation, the once I per se it to create very complex 3D worlds, that I can navigate through, and interact with while playing, using OSC messages mapped to my feature extraction system. Developing the software wold have to be done in windows, but I wold keep play with it on MacOSX.

Something I wold like to try wold, into a certain extent, be playing with more complex forms of analysis and mapping. I think that if I cold implement innovative means of artificial intelligence, like recursive swarms of hybrids of BDIs and artificial neural networks, my work, cold, into a certain extent, project into another dimension of analysis and mapping. Although considering to implement such an approach should be motivated, into a certain extent, not by some sort of technological fetiches, but rather as a mean of approaching problem solving, so if I need to optimize, for an instance, the way as I deal with gesture recognition or something like that, I can into a certain extent consider to implement such a model, otherwise it is just pre loss of time.

Another thing that I feel into a certain extent, that can be improved is the issue of Interactivity. The way I feel I should do it is by mapping basic synthesis routines with ugens to my feature extraction data, and map it to the several parameters of the things that I am mapping. Summing to this, inversions, interpolations, and gating of parameters. I think that if do so, I can have an interesting relationship between gestural control and mappings, and still have a control that does't follow a logic of one-to-one mapping, but rather, something a bit more indirect.

I don't feel the need of using massive arrays of speakers like WFS. This as to do with pragmatical reasons of most of my performance situations end up by taking place in stereo setups. So I feel like a high order ambisonic implementation in SC3 is all I need.

Another thing that I have been remembering for a long time is getting, in occasional situations, is trying to get a chance to play with robotic systems such as disklaviers and these sort of things.

One of the biggest mistakes I committed when trying to find a sense of unity within my work during the last two years was, into a certain extent, not limiting myself. I am an electroacoustic composer and sound artist as well as an improviser. I was trying to bring externals influences that were already appearing in my mdic as it was developing. Bt into a certain extent, instead of trying to limiting myself to these two genres of music (electroacoustic and free improvised), and letting the external influences I tried to pick p each one of these topologies of mdic and treat them separately, by trying to figure strategies in tape composition that I cold apply to live electronics music. During much time, I played with another genres of music and during the predeceasing time to come to sonology (I mean, the two or three years before coming to sonology), some of those elements start to become evident on my mdic. Limiting myself to viola, electronics and in a near ftre vocals (as an improviser viola player, and vocal performer), is the way I feel I should do this.

Getting what someone can call a personal voice or personal musical identity, is something that can take shorter and can take longer. Shouldn't be forced. Rather just committing to an acts sense of self awareness, and self criticism, that can help to develop the work. Some people have that from the beginning. Some people get aware of that as their work develops. It's pretty clear that my work as an improvisor has a strong sense of coherence and identity. So I think that the fact that I limited myself to the viola and to the laptop when improvising really made me get to a point in which by practicing, but at the same time having a sense of self awareness and self conciseness, with a specific goal in terms of what I was seeking, sometimes also learning the things that naturally happened in an unexpected way, was the reason of my development and language acquisition as an improviser.

I am an improviser soloist and chamber musician. and an electroacoustic music and sound artist. Merging all the aspects of my mdic while an improviser with viola and live electronics, is merely an issue of creating an instrument, that can process and behave adaptively to any kind of mdic material, and doesn't force me to adapt my way of playing. this is an issue of the synthesis within itself. At the same time the mappings and instantiation algorithms.

Getting it in a structured improvisation framework is easier than free improvised way. Because in structured improvisation, I can preview what kind of materials will I play, and play either a combination of real-time sampled and prerecorded material, and real time processed sons. Defines spatialisation routines, algorithms for controlling visuals, etc.

I need to have a space for making tape and interactive mdic. I need to have a space for making solo and collective improvisation. With electronics, without electronics. All the possible combinations of it. Structured and free, etc., sometimes having a single paradigm in one piece, sometimes having multiple. Develop approaches to form counterpoint and orchestration.

Intuitively, while improvising, while composing. Seeking, but not expecting to find results. They will come naturally. Limiting and from there expanding. Without excluding approaches. Both in my electronic and electroacoustic music and improvised all the things that I listen and do will come to if I give space for them to do so.I effectively believe that If I limit myself to electroacoustic mdic and free improvisation, my whole ontological content, revealed by my influences, the things I like, I don't like, the mdic that I enjoy listening to will come p in the middle of all of this. Most of the times it is an issue of experimenting having a goal, without the need to achieve complexity immediately, or having clearly defined rhetorical systematic schemes on how my mdic should, into a certain extent develop. It's just about breathing mdic and letting everything come together, so that I can into a certain extent, keep my ornery on self-knowledge and personal development the mdic. If I do so every day, I am sure that everything will come up naturally for results both with getting a unified personal language, and indeed, that my mdic has been developing over the corse of time, even remembering that I still, effectively have much to learn and to grow.

There are two sorts of feature extraction parameters: the ones that are directly related to physical parameters in the mdic both for sons and movement, captured and analyzed by all sorts of means; on another hand, the ones that are innerly connected to the nature of the mdic itself, for analysis of musical content. Examples of the first one can, effectively be, things like spectral centroid, brightness, loudness, noisiness, frequency or amplitude (even remembering that amplitude and loudness are much the same). On another hand we have parameters like the cue of new phrases or sections, the degree of rhythmical density, periodicity or regularity. Even, into a certain extent, things like pitch consonance, pitch class average, registry variance, etc.

Hyper-Viola	Target in terms of development after masters
Featre extractionhosted in	SC3 sing fft and recursive swarms of hybrids of artificial neural networks with bdis
Mappings done with	SC3 sing fft and recursive swarms of hybrids of artificial neural networks with bdis
Tools for sond processing and sampling	SC3 (with OpenCL), Kyma sing advanced forms of granlar synthesis, phase vocoding, sampling, catart, spectral mosicing, and another forms of complex synthesis methods; complex feedback strctres, physical models (eg, karpls strong), complex convoltion and spectral morphing techniqes, etc.
Tools sed for sond spatialization	SC3 with OpenCl sing High Degree of Ambisonics
Tools sed for visals	uDKOSC
Hyper-bow	
Platform of Development	SC3 (with OpenCL), Kyma
Sensors	KeithMC Millen Kbow, Softstep
Software sed for collecting sensor data	SC3
Hyper-composition	
Tools sed	SC3 (with OpenCL), Kyma

6. Bilbiographical references

Eigenfeldt, A. , 2007. Real-time Composition or Computer Improvisation? A composer's search for intelligent tools in interactive computer music, Simon Fraser University

Cadoz, C., Lisowski, L and Florens, J. L., (1990) A Modular Feedback Keyboard Design. Computer Music Journal, 14/2, MIT Press,

Chadabe, J., (1996), Electric Sound – the past and promise of electronic music. Prentice Hall, NJ, 1997. Chafe, C. D. and O'Modhrain, M. S. Musical Muscle Memory and the Haptic Display of Performance Nuance. Proceedings of the ICMC International Computer Music Conference. pp. 428 – 431

Chadabe, J. 1997. Electric sound: The past and promise of electronic music. Upper Saddle River, NJ: Prentice-Hall.

Chadabe, J. Interactive (1989) composing: An overview. In C. Roads, editor, The Music Machine: selected readings from Computer Music Journal, pages 143–148. MIT Press, Cambridge-London,.

Chadabe, J. (1977) "Some Reflections on the Nature of the Landscape within which Computer Music Systems are Designed".

Chadabe, J. (1980) "Solo: A Specific Example of Realtime Performance".

Chadabe, J. (1983) "Interactive Composing: An Overview",.

Chadabe, J. (2007) "Electronic Music: Unsung Revolutions of the 20th Century" http://www.percontra.net/6music2.htm,.

Chafe, C., B. Mont-Reynaud, and L. Rush. 1989. Toward an intelligent editor of digital audio: Recognition of musical constructs. In The music machine. Ed. C. Roads. Cam- bridge, MA: The MIT Press.

Clarke, E. 1985. Structure and expression in rhythmic performance. In Musical structure and cognition. Ed. P. Howell, I. Cross, and R. West. London: Academic Press.

Clarke, E. 1987. Levels of structure in the organization of musical time. Contemporary Music Review 2:211–238.

Clarke, E. 1988. Generative principles in music performance. In Generative processes in music. Ed. J. Sloboda. Oxford: Clarendon Press.

Clarke, E. 1999. Rhythm and timing in music. In The psychology of music. 2nd ed. Ed. D. Deutsch. London: Academic Press.

Cook, N. 1987. A guide to musical analysis. Oxford: Oxford University Press.

Cook, N. 1998. Analysing musical multimedia. Oxford: Clarendon Press.

Cope, D. 1990. Pattern matching as an engine for the computer simulation of musical style. In Proceedings of the 1990 International Computer Music Conference. San Francisco: International Computer Music Association.

Cope, D. 1991. Computers and musical style. Madison, WI: A-R Editions, Inc.

Cope, D. 1993. A computer model of music composition. In Machine models of music. Ed. S. Schwanauer and D. Levitt. Cambridge, MA: The MIT Press.

Cope, D. 2005. Computer Models of Musical Creativity. Cambridge, Massachusetts: MIT Press.

Miranda, E. R., and J. A. Biles, eds. 2007. Evolutionary Computer Music. London: Springer-Verlag.

Collins, N. (2006) Handmade Electronic Music: The Art of Hardware Hacking. Routledge, March

Collins, N. 2009. "Form and Algorithmic Composition." Contemporary Music Review 28(1): 103–114.

Collins, N. 2011. "SCMIR: A SuperCollider Music In- formation Retrieval Library." In Proceedings of the International Computer Music Conference. Available online at www.sussex.ac.uk/Users/ nc81/research/scmir.pdf. Accessed May 2012.

Collins, N. (2012), Automatic Composition of Electroacoustic Art Music Using Machine Learning, Computer Music journal, MIT Press journals

R. Dean. (2003) Hyperimprovisation: Computer-Interractive Sound Improvisations. AR Editions, Middleton, Wisconsin.

Eigenfeldt, Arne (2007) "The Creation of Evolutionary Rhythms within a Multi-agent Networked Drum Ensemble"

Freed and Uitti, (2006). Augmenting The Cello. NIME Proceedings

Garnett, Guy (2001) "Aesthetics of Interactive Computer Music".

Griffith, N., and P. Todd, eds. 1999. Musical networks: Parallel distributed perception and performance. Cambridge, MA: The MIT Press.

Impett. (1994) A meta-trumpet(er). In Proceedings of the International Computer Music Conference, pages 147–149, San Francisco,. International Computer Music Association.

Impett (1998) The identification and transposition of authentic instruments: Musical practice and technology. Leonardo Music Journal, 8:21–26,.

Jensen, E. Jensen Electric Cellos, 2006, http://www.halcyon.com/jensmus/cello.htm.

Jehan. (1997) Music signal parameter estimation. Master's thesis, IFSIC, Rennes, and CNMAT, Berkeley, September.

Jehan. (2001), Perceptual synthesis engine: an audio-driven timbre generator. Master's thesis, MIT Media Laboratory.

Jehan. (2004) Perceptual segment clustering for music description and time- axis redundancy cancellation. In Proceedings of the 5th International Conference on Music Information Retrieval, Barcelona, Spain, October.

T. Jehan. (2005) Hierarchical multi-class self similarities. In Proceedings of IEEE Wokshop on Applications of Signal Processing to Audio and Acoustics (WASPAA), Mohonk, NY.

T. Jehan, T. Machover, and M. Fabio. Sparkler: An audio-driven inter- active live computer performance for symphony orchestra. In Proceedings International Computer Music Conference, G ¨oteborg, Sweden, 2002.

T. Jehan and B. Schoner. (2001) An audio-driven, spectral analysis-based, per- ceptual synthesis engine. Journal of the Audio Engineering Society.

Jehan, Tristan. 2005. Hyperviolin. [Online]. Available from: http://web.media.mit.edu/~tristan/Projects/hyperviolin.html.

Lewis. (2000) Too many notes: Computers, complexity and culture in Voyager. Leonardo Music Journal, 10:33–39.

Lewis, George “Interacting with Latter-Day Musical Automata”, 1999.

Minsky, M. (1985) A Framework for Representing Knowledge. In Readings in Knowledge Representation, edited by Ronald J. Brachman and Hector J. Levesque. Los Altos, Calif.: Morgan Kaufmann Publishers, Inc.

Miranda, E., ed. 1999. Readings in music and artificial intelligence. The Netherlands: Harwood Academic Publishers.

E. Miranda and M. Wanderley. (2006) New Digital Musical Instruments: Control And Interaction Beyond the Keyboard. AR Editions, Middleton, Wisconsin.

Paradiso, (1997) J. New Ways to Play: Electronic Music Interfaces. IEEE Spectrum 34/12, cover article and pp.18-30, December

Paradiso, J. 1999. The Brain Opera technology: New instruments and gestural sensors for musical interaction and performance. Journal of New Music Research 28(2):130–149. Parncutt, R. 1988. Revision of Terhardt’s psychoacoustical model of the roots of a musical chord. Music Perception 6:65–94.

Palacio-Quintin, Cléo. 2003. The hyper-flute. Proceedings of the 2003 conference on New interfaces for musical expression. [Online]. SESSION Demo I. Pages 206-207. [Ref. 15 November 2006]. Available from: http://portal.acm.org/citation.cfm?id=1085764&coll=Portal&dl=GUIDE&CFID=64029 11&CFTOKEN=33409597.

Pope, S. T., and A. Kouznetsov. 2004. Expert Mastering As- sistant (EMA) Version 2.0: Technical Documentation. FASTLab Inc. Product Documentation. Center for Research in Electronic Art Technology, University of California Santa Barbara.

Puckette, M. S., Apel, T. and Zicarelli, D. D., (1998) Real-time audio analysis tools for Pd and MSP. Proceedings of the International Computer Music Conference (ICMC), Raaijmakers, D. Cahier 'M', (2000) A Brief Morphology of Electric Sound. Orpheus Institute, Gent Belgium

Roads, C. 1985. Composers and the Computer. Los Altos, California: William Kaufmann.

Rowe, R. 2001. Machine Musicianship. Cambridge, Massachusetts: MIT Press.

Rowe, R. 1993. Interactive music systems: Machine listening and composing. Cambridge, MA: The MIT Press.

Rowe, R., and T. Li. 1995. Pattern processing in music. In Proceedings of the fifth biennial symposium for arts and technology. New London, CT: Connecticut College.

Rowe, R., and E. Singer. 1997. Two highly integrated real-time music and graphics performance systems. In Proceedings of the 1997 International Computer Music Conference.

R. Rowe. 1992 Interactive Music Systems. MIT Press,.

Rowe, R. (1993) Interactive Music Systems: Machine Listening and Composing. Cambridge, MA: The MIT Press.

Tanaka, 2006 editors, NIME, pages 277–282. IRCAM - Centre Pompidou in collaboration with Sorbonne University,.

Trueman, D. (1999), reinventing the violin “the infinite violin: the descontructed violin reconstructed”, Princeton university

Uitti, F.-M. Two Bows, 2006, http://uitti.org/twobows.html.

Vaggione, H. 1984. “The Making of Octuor.” Computer Music Journal 8(2):48–54.

M. Waisvisz. 1985 The hands, a set of remote midi-controllers. In Proceedings of the International . International Computer Music Association.

M. Wanderley. 2003 Quantitative analysis of non-obvious performer gestures. In Gesture and Sign Language in Human-Computer Interaction: International Gesture Workshop, pages 241–253,.

Volker Krefeld (Summer, 1990); Michel Waisvisz, Computer Music Journal, Vol. 14, No. 2, New Performance Interfaces 2. , pp. 28-33.

Wanderley, M. M. and Battier, M. (eds.), 2000 Trends in Gestural Control of Music. IRCAM, Paris,

Zorn, 2002 J.E. Arcana. Granary Books,.

performances
Sonology Electroacoustic Ensemble Performance with Peter Evans at Sonology Discussion Concert, Royal Conservatory, Den Haag
concert with Yedo Gibson and Ofir Klemperer at Studio Loos, Epemere, Den Haag
audiovisual solo performance of tape pieces, Ephemere, Den Haag
visuals for Steindor Kristinsson at Hyphae Festival, Den Haag
Sonology Discussion Concert Sonology Electroacoustic Ensemble Performance, Royal Conserevatory, Den Haag
Sonology Discussion Concert Performance of Audiovisual Tape Piece Arcana, Royal Conservatory, Den Haag
solo audiovisual performance of tape pieces at Fiber Festival, Amsterdam
live act with V4W.Enko at Fiber Festival, Amsterdam
Informatic Assistance for the piece first Approach to Morgenstra to Juan Alberracin, Royal Conservatory of Den Haag
concert with Anne La Berge, Leo Svirsky, and Renato Ferreira, Marthin Luther Kirk, Amsterdam
audiovisual solo performance of tape pieces, Puremagnetic, Cafe de Vinger, Den Haag
performance of a piece by Petra Strahovnik at Composition Concert, Royal Conservatory of Den Haag
performance of a piece by Corne Van Roos at Composition Concert, Royal Conservatory of Den haag
performance of a piece by Corne Van Roos at Composition Concert, Royal Conservatory of Den haag
performance of a piece by Petra Strahovnik at Korzo Theatre, Den Haag
informatic assistance to Klara Andriova final Exam, Michael Gordon XY piece, Royal Conservatory, Den Haag
villa kavilla, ambient fixed media shared performance with Nicolla Casetta
Villa Kavilla, Free Improvisation with Live electronics performance with Nicolla Casetta, Den Haag
Blik Opener Festival, Installation with fixed media pieces
Private concert with James Hewitt and Renato Ferreira, Den Haag
Informatic assistance for roel goedhart ohanna guitar concerto version for guitar, electronics, and visuals
performance at sonology discssion concert of my piece 33^12/16^12 for viola and live electronics
performance at cass concert of my piece 33^12/16%^12 tape version