UNIVERSIDADE FEDERAL DE PELOTAS

Centro de Artes

Curso de Composição Musical

Trabalho de Conclusão de Curso

Using Juggling as a Controller for Computer-Generated Music:

An Overview of the Creation of an Interactive System Between Juggling and Electronic Music

Gustavo Oliveira da Silveira

Pelotas, 2014

Gustavo Oliveira da Silveira

Using Juggling as a Controller For Computer-Generated Music:

An Overview of The Creation Of An Interactive System Between Juggling And Electronic Music

Trabalho de Conclusão de Curso apresentado ao Centro de Artes da Universidade Federal de Pelotas, como requisito parcial à obtenção do título de Bacharel em Composição musical.

Orientador: Professor Dr. James Correa Soares

Pelotas, 2014

Gustavo Oliveira da Silveira

Using Juggling as a Controller For Computer-Generated Music:

An Overview of The Creation Of An Interactive System Between Juggling And Electronic Music

Trabalho de Conclusão de Curso aprovado, como requisito parcial, para obtenção do grau de Bacharel em Composição Musical, Centro de Artes, Universidade Federal de Pelotas.

Data da Defesa: 18/12/2014

Banca Examinadora:

.....
Prof. Dr. James Correa Soares, Doutor em Composição pela University at Buffalo.

.....
Prof. Dr. Jorge Meletti, Doutor em Composição pela Universidade Federal do Rio Grande do Sul.

.....
Prof. Dr. Rogério Constante, Doutor em Composição pela Universidade Federal do Rio Grande do Sul.

Agradecimentos

À minha mãe, pelo amor e apoio incondicional, por sempre ter me incentivado a seguir os meus sonhos e investido em mim não importasse as dificuldades.

Ao meu pai, por ter me colocado no mundo da música, por ter me ensinado as primeiras canções e por sempre ter me apoiado e incentivado a seguir fazendo o que eu amo.

Ao Hugo Azevedo, que apesar de não compartilhar o mesmo sangue é família desde que nasci, por apoiar sempre a minha mãe, a mim e pelo bom humor.

Aos amigos pra vida toda que fiz durante a graduação, pelos ensinamentos, pela música e mais música, pelas farras, pelo apoio, por toda a felicidade que me proporcionaram.

Ao meu orientador e professor James Correa e aos demais professores que tive, pelos ensinamentos, por terem aberto a minha mente para um mundo que eu não conhecia.

À Carolina Brum, pela ajuda inestimável para com este TCC, pelos ensinamentos e pelo carinho.

To professor Martin Herman, from CSULB, for his amazing classes and for helping me in the beginning of this research.

Ao diretor João Bachilli e aos artistas do Grupo Tholl, por terem me dado ensinamentos que nenhuma faculdade poderia me dar.

Obrigado!

“Any sufficiently advanced technology
is indistinguishable from magic.”
(Arthur C. Clarke)

Abstract

SILVEIRA, Gustavo Oliveira da. **Using Juggling as a Controller For Computer-Generated Music:** An Overview of The Creation Of An Interactive System Between Juggling And Electronic Music. 2014. Arts Center, Federal University of Pelotas, Pelotas, 2014.

This work presents aspects that concern the relationship between juggling, programming and music, in the creation of an interactive system for generating and controlling electronic music through a juggling routine. It will be discussed the reasons for choosing juggling balls as controllers, and also, the reasons for choosing the software Max, Ableton Live and Max For Live to map juggling and music. Developing an interactive system between these areas suggests new paradigms in the creation of a juggling routine, as well in the creation of a music composition. Throughout the years of acoustic musical instruments, we know that their physical features imply compositional choices. The same happens with New Interfaces For Musical Expression (NIME) design, that is, the physical features of the juggling balls influence the musical composition. As a result of this research, it is presented some of the possibilities and difficulties while working with this material as a controller for computer-generated music. Finally, it reports how this NIME design affected my ways of thinking juggling and composition.

Key-words: Interactive system; juggling; computer-generated music; music juggling controller; NIME.

Resumo

SILVEIRA, Gustavo Oliveira da. **Usando Malabares Como um Controlador Para Música Gerada por Computador:** Uma Vista Geral da Criação de Um Sistema Interativo Entre Malabares e Música. 2014. Centro de Artes, Universidade Federal de Pelotas, Pelotas, 2014.

Este trabalho apresenta aspectos que dizem respeito à relação entre malabares, programação e música, na criação de um sistema interativo para gerar e controlar música eletrônica através de uma rotina de malabares. Serão discutidas quais são as razões para se escolher bolas de malabares como controladores, e também, as razões para se escolher os programas Max, Ableton Live e Max For Live para mapearem malabares e música. Desenvolver um sistema interativo entre essas áreas sugere novos paradigmas na criação de uma rotina de malabares, assim como na criação de uma composição musical. Ao longo dos anos de instrumentos musicais acústicos, nós sabemos que suas características físicas implicam em decisões composicionais. O mesmo acontece com o design de Novas Interfaces Para Expressão Musical (NIME), isto é, as características físicas das bolas de malabares influenciam na composição musical. Como resultado dessa pesquisa, são apresentadas algumas das possibilidades e dificuldades ao se trabalhar com este material como controladores para música gerada por computador. Finalmente, é retratado como estes resultados afetaram a minha maneira de pensar em como criar uma rotina/composição para este NIME.

Palavras-chave: Sistema interativo; malabares; música gerada por computador; malabares como controlador musical; NIME.

Summary

1 Introduction

Trough the years that I have been working with music, as a composer or performer, I came across with other several types of art. These arts include: visual arts, cinema, theater and circus. While I was in contact with other fields, my interest in creating multidisciplinary works had grown, so, I started learning tools from computer programming and visual arts. For about seven years, I worked as composer, musical director, and instrumentalist of a circus group. Group members of this company are encouraged to take all sorts of classes, such as dance, theater, music, juggling, yoga, etc. The only one that I felt safe and absolutely fun was juggling. Since then, I have been practicing it, going to juggling festivals and networking with other jugglers. Juggling became a serious hobby for me.

During Spring 2013, while studying the discipline “Literature and Aesthetics of Interactive Media”, in the California State University – Long Beach, thought by Dr. Martin Herman, I had to write an article of any subject related to the discipline. At that time, I already had some experience with computer programing and decided to try to accomplish an old desire, creating a software that would integrate juggling and music. I started creating a system that would make juggling and music one connected device, where music would respond in real time to what the juggler would do, leading to a juggling musical instrument.

Working with juggling and music brought me the interest to build an interactive system between juggling and computer-generated music, adding the different qualities of both. But, before starting talking about the system, what is the concept of juggling?

> Juggling is the art of keeping objects in the air by throwing them from hand to hand in a controlled, rhythmical way. A very plain definition of juggling might be “keeping more objects in the air than you have hands. (AUSTIN apud WAGENAAR, 2009)

There are several types of juggling devices, like balls, rings, clubs, etc., but this work focuses only on juggling balls. A juggling ball has a uniform size

and movement in the air, and it doesn't spin like a club. So, for me, because of that, it is a good juggling device for being transformed in a music controller. Balls suits well the motion tracking strategy presented further in the text.

> It is important to note that, although a juggler can certainly improvise, he or she can not just throw objects into the air randomly - and *keep* juggling. What you see when a juggler is apparently freely ignoring the laws of gravity, is a thoroughly trained knowledge and intuition on the theoretical rules of juggling, that can make a juggler react so fast it looks like *everything* is possible. But this is not true: all throws, catches and body movements are subject to strict rules, and are possible only under certain conditions. These conditions can then be recognized and created while juggling. (WAGENAAR, 2009)

Juggling respects a lot of mathematical rules, as Wagenaar says. A juggler knows the kinds of throws he can do in order to fulfill a trick, not letting the balls collide with each other or fall from the juggler's hands. The mathematical rules that determine whether a trick[1] is possible or not is called *Siteswap*.

> "Site swap-notation was invented by Bruce Tiemann in 1985, and has been widely used by jugglers ever since (Tiemann and Mangnusson 1991; Juggling Information Service 1993). Site swap keeps track of the total number of objects in a pattern, very precisely indicates height and direction of the different throws in that pattern (that is, how objects 'swap sites'), but ignores all other parameters such as the kind of object, the number of hands, the general speed of throwing, and body movements (Juggling Information Service 1993, Dancey 1995). For a surprising percentage of tricks, this works very well. Site swap is a beautiful system that is able to capture a wide range of juggling patterns in a simple numeral notation, with only a few basic rules." (DANCEY 1995, WRIGHT 1996).

The details of the *Siteswap*[2] notation are not going to be presented in this work. For understanding the relation between the Siteswap with the system, it is just necessary to know that as high a throw is, the higher is the number in the *Siteswap*. For instance, a 5 throw, usually above the head, is higher than a 3 throw, at the chest height. So, supposing that the interactive

---

1 A trick is every sequence that a juggler can do repeatedly.

2 For more details about the *siteswap* notation visit the website http://www.siteswap.org, or watch the video Mathematics of Juggling (https://www.youtube.com/watch?v=38rf9FLhl-8).

system is based on how high the throws are, a 3 throw could trigger something, while a 4 could trigger another thing, and so on. Considering the interaction between juggling and music, Siteswap can be used not only as juggling notation, but also as a music notation/guide. The most common trick with three balls, the cascade, is made by throwing all the balls at the 3 height, noted like ”333”, just “3”, or how many “3s” it is wished to represent. A trick with a throw at 5, one at 3 and one at 1, is noted like “531”. A more complex sequence of movements could be notated like “3335314415151513333”, for example[3].

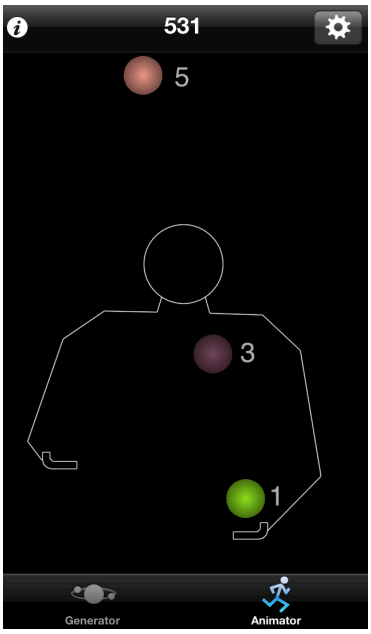


Figure 1. *Snapshot of the Siteswap “531”, extracted from the app “iJuggle”[4].*

[3] Siteswap also can prove, by a mathematical formula, if a trick is possible or not. In this case, this sequence is a possible sequence of throws.

[4] The iJuggle is an application that creates a 2D animation of the Siteswap notation. The app can be downloaded at: https://itunes.apple.com/us/app/ijuggle/id306936948?mt=8.

2 Building an Interactive System

2.1 Interactive Systems

Several features of interactive music systems are appealing and interesting to me, for instance: what an instrument can do, how are the systems built, and how gestures and sound are mapped.

> Interactive music systems are used in many different contexts including installations, net- worked music ensembles, new instrument designs and collaborations with robotic performers (Eigenfeldt and Kapur 2008). These systems do not define a specific style – that is, the same interactive model can be applied to very different musical contexts. (DRUMMOND, 2009)

As Drummond says, it is hard to define exactly what an interactive system is. Rowe (1993), in his book *Interactive Music Systems*, defines it as:

> Interactive computer music systems are those whose behaviour changes in response to musical input. Such responsiveness allows these systems to participate in live performances, of both notated and improvised music. (ROWE apud DRUMMOND, 2009)

Drummond (2009) mentions that the definition in Rowe's book focuses on the responsiveness of the system, where the system listens to the performer and generates the music. In Rowe's concept the influence of the music on the performer is secondary.

Alternatively, my juggling/music system suits the *Interactive Composing* concept, made by Chadabe (1997).

> These instruments were interactive in the same sense that performer and instrument were mutually influential. The performer was influenced by the music produced by the instrument, and the instrument was influenced by the performer's controls. (CHADABE apud DRUMMOND, 2009)

An important concept of this work is the requirement of having the interactive system using juggling and music in a way that one is influenced by the other. A working hypothesis is that the interaction would bring new possibilities in the creation of a juggling routine. This said, unusual gestures and throws, that might not be so interesting in a regular routine, can become

interesting with a interactive juggling and music system.

> Critical investigation of interactive works requires extensive cross-disciplinary knowledge in a diverse range of fields including software programming, hard- ware design, instrument design, composition techniques, sound synthesis and music theory. (DRUMMOND, 2009)

In order to design an interactive system between juggling and music, as mentioned by Drummond, one needs to have a cross-disciplinary knowledge in a diverse of fields. Gathering knowledge on juggling, music, and programming allowed me to trace parallels no only between juggling and music, but also to integrate them with a computer program.

2.2 The Software

One of my first concerns was on how I would translate the balls' action into computer data. I decided to use the software Max[5], from Clycling '74, for this purpose. During my undergrad studies, I was introduced to different types of programming environments, such as Max, Pure Data, Super Collider, Csound, Python, etc.. However, I just got deeper in Max and C++. Besides having previous experience with Max composing electroacoustic and interactive pieces, Max offers real-time audio and video capabilities, making it the ideal programming environment to build my software.

> "Max is a visual object oriented programming language which has three cores. The first core is the Max core which handles mathematic functions. The second core is MSP which is used for signal processing to generate sound and manipulate existing sound. The thirds core is Jitter which is used for video processing. All of the cores are fully accessible from application which makes Max a very powerful multimedia visual language." (BERG, 2012)

After choosing Max to be the interface between juggling and music, I had to decide how the music would be generated. Although all the interactive system could be done in Max, I decided to work with another software too,

5 https://cycling74.com/products/max/

called Live[6], from the company Ableton. Live is a great digital audio workstation that allows one to easily work with loops and effects. Live also includes a tool called Max for Live (M4L), which bridges Max and Live. M4L allowed me to make things that I only could do on Max, but with the flexibility of Live.

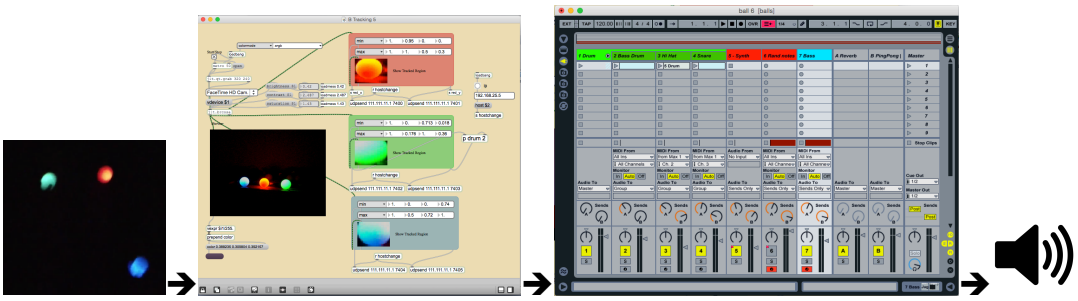


Figure 2. *How the system flows: Juggling routine ➔ Motion Tracking inside Max ➔ Music generation inside M4L and Live ➔ Speakers. The details of each part will be discussed later on.*

2.3 The Hardware

The hardware design starts by selecting the sensor technologies to track the balls' motion. Some of the options were camera, laser sensors, and contact microphones on the juggler's hands. I chose to use the camera, as it would allow me to continuously track each of the balls individually, instead of a discrete tracking provided by the other methods. Another advantage in using a camera is that it allows one to track several parameters at the same time in a juggler's routine, depending on how many balls are used.

The type of ball was chosen taking into account how the motion tracking works. To track each ball separately it is needed that each one has a different color. The motion tracking works on tracking specifically colors, so, for this purpose, glowing balls showed to be the most reliable devices. Just having glowing balls in the dark, without a background or other colors that

6 https://www.ableton.com

might be tracked by mistake, makes a safer way to work with the motion tracking system.

In the next chapters it will be discussed the techniques and concepts used to build the software, the music issues involving in developing the interactive system, the hardware used, and the main objects[7] in the patch[8].

[7] An object in Max/MSP is a box where you can program its function by writing a word, or text, where it can be connected to other boxes in a signal flow.

[8] A patch is a program created inside Max.

2.4 The Max Patch

2.4.1 An Overview of the Patch

The Max patch is responsible for three things: tracking the balls' positions in the air, mapping it to numbers, and sending it to Live. Then, these are used as musical parameters.

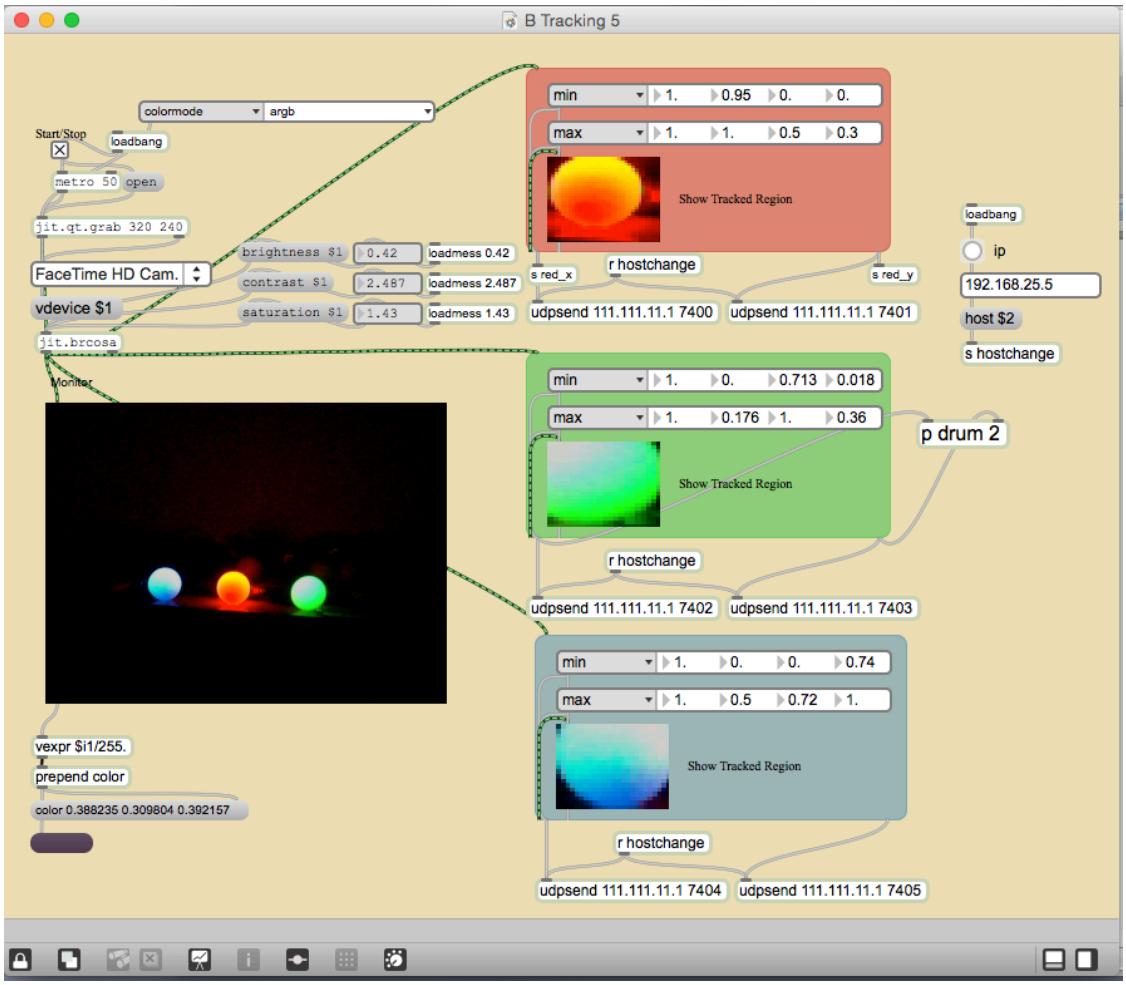
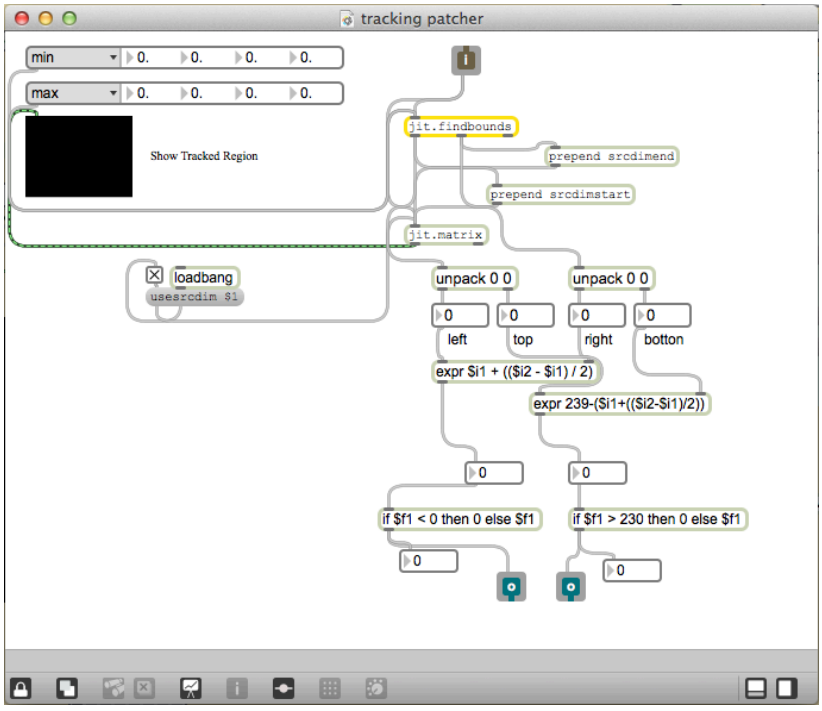


***Figure 3**. The main window of the Max patch. In the left it can be seen the monitoring of the camera and in the right three boxes, each one monitoring each ball separately.*

2.4.2 Motion Tracking

For this work the main object used in the motion tracking was "*jit.findbounds"*. The description for this object in the software's help file is: "Locate bounding dimensions for a value range. Scans a matrix for values in the range [min, max] and sends out the minimum and maximum points that contain those values. The minimum point is sent as a list out the leftmost outlet, and the maximum point as a list out the second outlet. If both points are -1 values, then there are no points within the range." This object creates a quadrilateral around the tracked object, giving the position value of each bound. Then, though some calculations, one can have the value of the exact center of the tracked object. For tracking different colors using *jit.findbounds,* this Max object accepts the minimum and maximum values of alpha, red, green and blue (RGB) for the desired hue, ranging from 0 to 1. As a consequence, the RGB color ranging form 0 to 255 should be mapped to the scale [0,1]. This is made in a *bpatcher*[9] in the main patch, as shown in figures 4 and 5.



9 The Max help file says that a *bpatcher* "Abstracts the content of a patcher or subpatcher for use in other patchers, displaying only those visual elements which are desired".

Figure 4. *Bpatcher that allows you to control the range of RGB for tracking each ball.*

As seen in Figure 4, the yellow object, “jit.findbounds”, is connected to the “jit.brcosa” object through the patcher’s inlet[10]. The “jit.brcosa” allows one to control the image brightness/contrast/saturation. Also, “jit.findbounds” receives information from the “attrui” object, in the upper left (Figure 3), that inspects the attribute values of the object that it is connected to, allowing the selection of minimum and maximum of red, green and blue, as shown in Figure 5. These values are used to track the desired color.

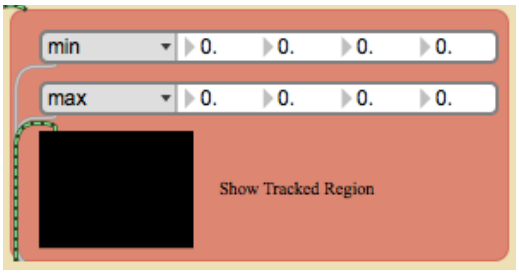


Figure 5. *Abstraction of the Bpatcher in figure 4.*

2.4.3 Sending the Information to Ableton Live

The initial approach was to use midi as data type to communicate with Live, because Live has an easy midi mapping system. However, it turned out that this mapping was designed for commercially available devices. Given the lack of flexibility to other hardware interfaces, I decided to try Open Sound Control (OSC)[11] protocol.

---

[10] The Max help file says that an Inlet can be used within a patcher to receive messages from outside the patcher wherever it is embedded. Each inlet object in a patcher will show up as an inlet at the top of an object box when the patch is used inside another patcher (as an object or a subpatch). Messages sent into such an inlet will be received by the inlet object in the subpatch. A patcher can have a maximum of 250 signal inlets. The number of data inlets is a much bigger number than that.

[11] “Open Sound Control (OSC) is a protocol for communication among computers, sound synthesizers, and other multimedia devices that is optimized for modern networking technology. Bringing the benefits of modern networking technology to the world of electronic musical instruments, OSC's advantages include interoperability, accuracy, flexibility, and enhanced organization and documentation.
This simple yet powerful protocol provides everything needed for real-time control of sound and other media processing while remaining flexible and easy to implement.” This information

OSC seemed to be a great tool to bridge Max and Live, however, Live doesn't have an OSC mapping feature. The solution, then, was to use M4L. Two objects are used to send OSC information: "udpsend" and "udpreceive". They require two kinds of information: the IP address and the port that it will communicate through. Each of the three balls has two values – one for Y-axis and one for the X-axis – resulting in six values sent from Max to Live through six OSC ports.

2.5 The Music Engine

2.5.1 Concerns About Creating a Juggling/Music Instrument

Creating a system to track the ball's motion is technically a simple task to do, however, translating this information to an aesthetically interesting juggling/music routine turned to be more difficult. There will be always two values for each ball, the X and Y position in a 320x240 pixels space. But, what to do with these values? Before deciding the overall sound, timbre, style of music that I would like to have, many tests were made. I created different modules, each module being responsible for creating a different layer in the final musical result, each layer working as a different instrument. At first, I did some simple synths that could work as "melody layers". The first one was a sine wave, using the object *cycle~*, with a phase modulation. The Y-axis controls the pitch in the range [20, 1000] Hz, where the lower frequency is in the bottom region tracked by the camera. The phase is controlled by the X-axis, ranging from 0 Hz to 30 Hz. Also, the X position controls the panning, so, the viewer can have the sensation that the sound is coming from the ball. This way, using this synth in the interactive system can be a clear way to show that the synth is being controlled by the balls' motion. The second one is very

---

was taken from the website www.opensoundcontrol.org, accessed in November 25th, 2014.

similar, however, the X position controls the pitch of a sawtooth wave whereas the Y position controls a filter. Both mappings can be found in figure 6.

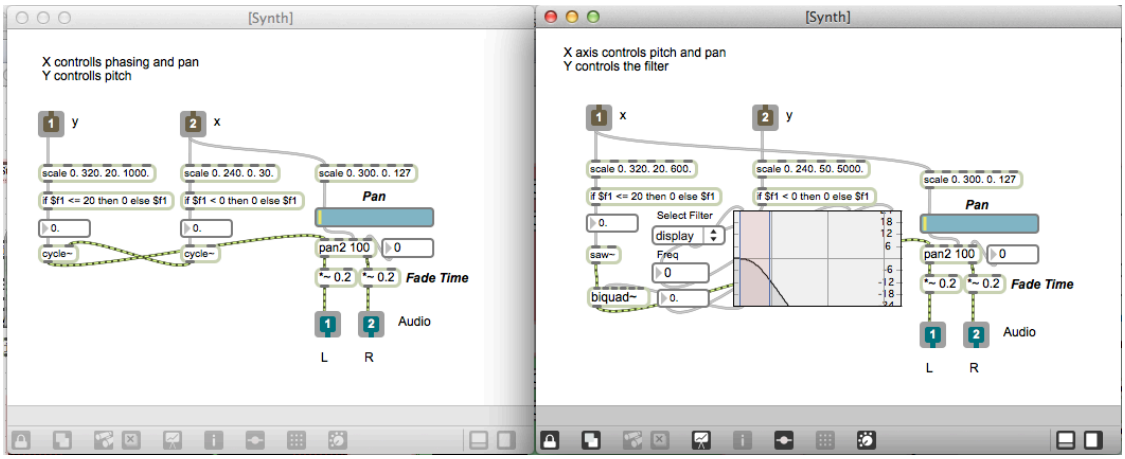


Figure 6. *The first two modules created.*

These modules were interesting controlling one or two balls at time, however, when adding a third one the result was not as good as before. When using three balls, in a “333” Siteswap sequence, for example, the balls’ movement are continuous, making the system to generate continuous *glissandi*, leading to a repetitive sound. This repetition doesn’t suit what I intend for my routine. Juggling, just as music, can have a strict sequence of movements or it can be improvised. For my routine, I wanted a music that sounded improvised, but related to my movements somehow, that the audience could relate movement with sound. So, assigning the position of a ball directly to the pitch would not work for three balls at the same time.

The sound of three balls doing *glissandi* doesn’t work well for what I want, but, juggling is not restricted to three or more juggling devices. At this point in my research, I realized that I was too attached to what I knew as juggling tricks and decided to go some steps back in my juggling technique, experimenting with just one or two objects. For my work, building an interactive system between juggling and music is not just related to tracking a juggling routine and mapping it to music. It is, indeed, the creation of a NIME (New Interface For Musical Expression), requiring the discussion of the intersections between music and juggling, opening possibilities for changes on my music and juggling practice. Keeping that in mind, I tried to use the

juggling balls as they were equally important as juggling device and as music instrument. For example, moving one ball very slowly in the air might not be very exciting to watch in a juggler's routine, but, when its movement is controlling the pitch of a synthesizer, it achieves an interest that a regular juggling ball would not have. The opposite is the same, perhaps watching a violinist playing a very slow *glissando* is not exciting either, but listening this sound being made by a glowing ball, with no wires, can be more interesting by its peculiarity and visual appeal.

Using one or two balls allows clear relations between juggling and music, like controlling a *glissando*, for example. For three balls, however, the balls' positions are mapped to an improvisation set. This will be clarified further on.

The desired mapping would make the juggler feels he/she was being accompanied by a live band, that would interact with his juggling routine. In order to obtain this relation, I would have to create one different module for each ball. The modules would have to "improvise" notes that would work as the rhythmic, harmonic/melodic, and bass sections. Each instrument would "improvise" in its own way. This would create a tradeoff between a clear relation between music and the balls' movements and a more appealing music outcome.

These paradigms brought by the NIME's creation reveals the need for a new kind of performer: the "musician juggler" or the "juggler musician". This performer will, probably, be closer to one of the practices only. But, one will need to have both qualities for making a good routine with this device. Good musical qualities will not guarantee good visuals, as well as good juggling qualities will not guarantee good music. Therefore, a juggler would need to know music, as well as a musician would need to know how to juggle. Throughout the years, most of the jugglers that I have met had a very strong musicality. Some of the jugglers were musicians before they started to juggle, like me. For several reasons, juggling is closely related to music, due to its very rhythmically patterns, that can easy be related to music, for example. This relation can motivate musicians to learn how to juggle and jugglers to

learn how to make music. So, building a device that is both juggling and music instrument seems to be a good idea for those that work with both arts.

2.5.2 The Ableton Live Project

In this section, it will be discussed the implementations using Ableton Live; its role in the music engine; and how the project is organized to work with the system.

2.5.2.1 An Overview of the Project

Inside Live, some tracks were created, each one working as a layer of the musical outcome, similarly to an instrument in a band.

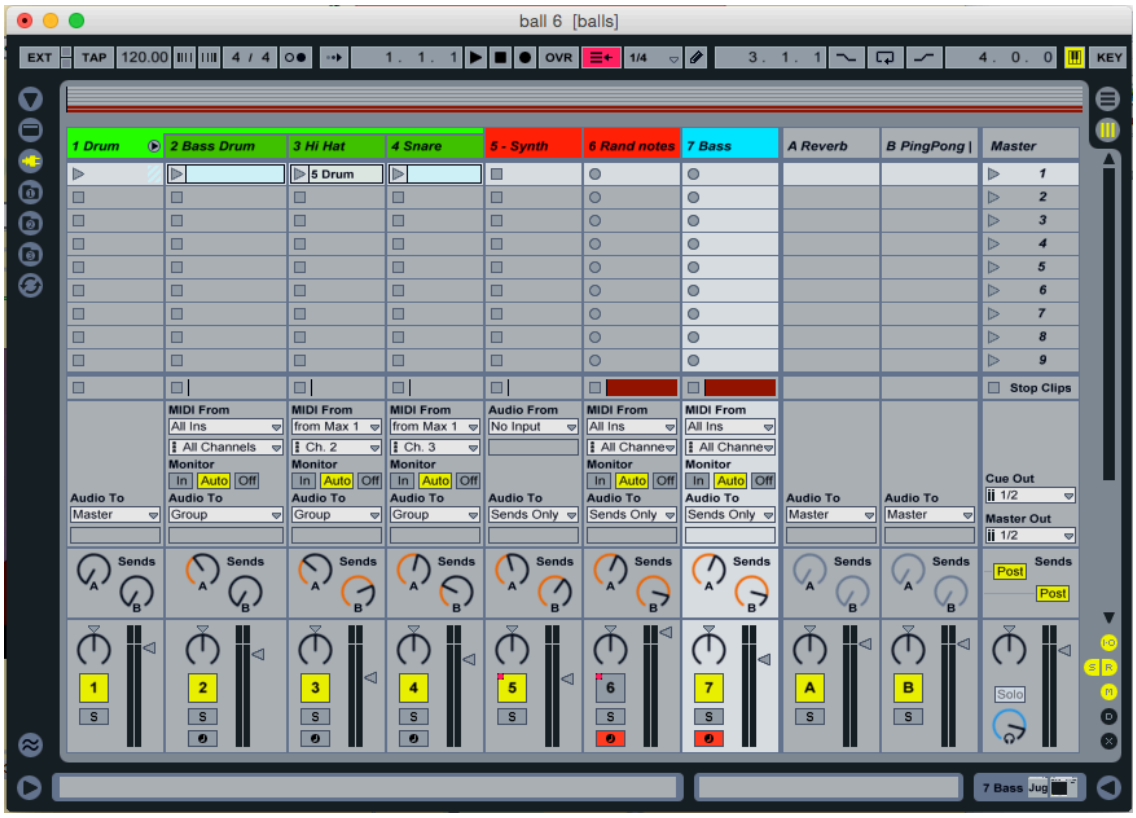


Figure 7. *The Ableton Live Project.*

There are four main tracks, “Drum” – with three sub tracks – “Synth”, “Rand notes”, and “Bass”. Also, there are two “send and return” tracks, one with reverb and one with delay effects. One great advantage of using Live as the music engine is its simplicity on working with all tracks in the same tempo.

In the upper left corner of Figure 7, the tempo chosen was 120 bpm, because it fits the average speed of the juggling movement, making it easy to match the throws with the beats. All tracks receive information from Max with the object “udp.receive”, or via MIDI, and its information controls a diversity of parameters in Live, such as triggering loops, controlling filters, scales, register, etc.

2.5.2.1 The Melody Section

The melody section was the first to be built – in Max – and was later transferred and improved to run on M4L. There is a very clear connection between the ball’s movement and its sound. This module sounds very similar to a Theremin, where the pitch is controlled with vertical motions. The sound is created inside M4L using a sine wave, whose phase is controlled by another sine wave. One oscillator, the carrier, receives data from the Y position of the ball, controlling its frequency, while the other oscillator receives X data, controlling the phase of the carrier. The signal of the carrier is sent to the send and return channels, where reverb and delay effects are applied. Also, the X position controls the panning of the sound.

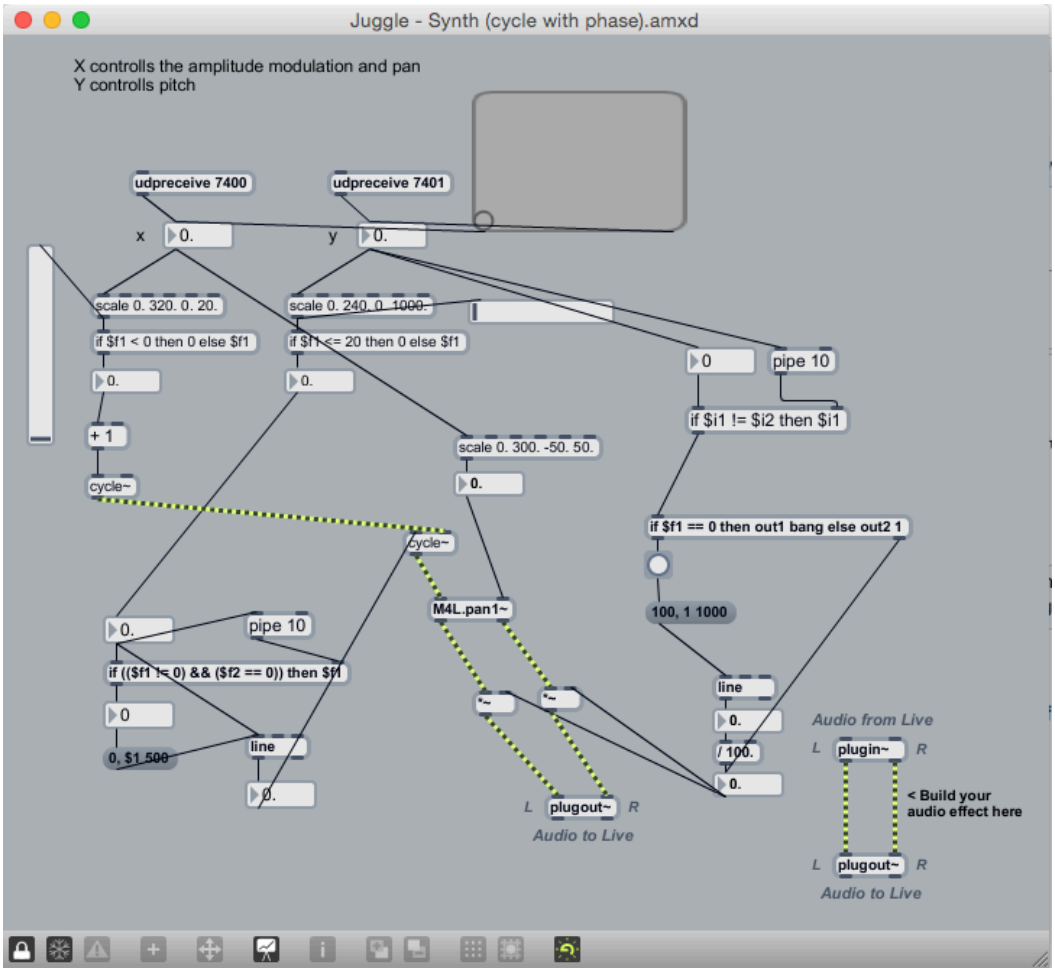


Figure 8. *The melody synth.*

2.5.2.2 The Harmony Section

For the harmonic part, I intended to have the sensation that a piano player was improvising in a certain scale, giving a harmonic foundation. Although it would sound like an improvisation, it had to have a relationship with the ball's movement. In order to obtain this relation, I decided to control the following parameters: the register, the speed, and the scale used by the "improviser".

To program this "improviser", several parts were necessary: a random rhythm generator, a random note generator, a scale selector, a register selector, and a generator of random velocities and durations. All the

information would be sent to another synth in M4L, or to a virtual instrument in Live. The mapping works as the follows. The ball's height – given by the Y position – is mapped to the octaves played by the improviser, e.g., a higher position is mapped to higher octave. Also, the Y-axis is divided in three regions, which are related to the note triggering speed, i.e., the bottom region triggers the slowest rhythms, whereas the other two upper regions trigger the faster rhythms. The X-axis is also divided in three regions that are mapped to three scales.

The information flow can be seen in Figures 9, 10, and 11. The patch receives the data from "udp.receive", and it is, then, scaled to be sent to a group of objects. The X data is processed and sent through "s scale" which is transmitted to "r scale" to select one out of three **scales** (Figure 12)**:** natural minor, harmonic minor, and whole tone. The Y data is processed and sent through "s octave" which is transmitted to "r octave" to select one out of three **octaves**. The camera resolution is 320 x 240. This resolution is scaled down to generate the three intervals in each axis.

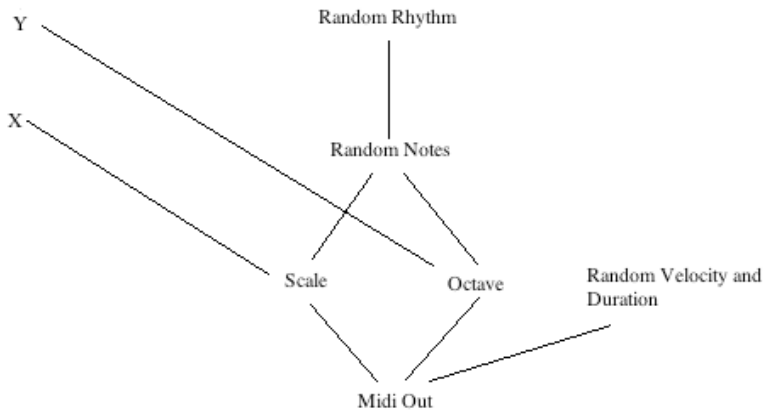


Figure 9*. The flow of information.*

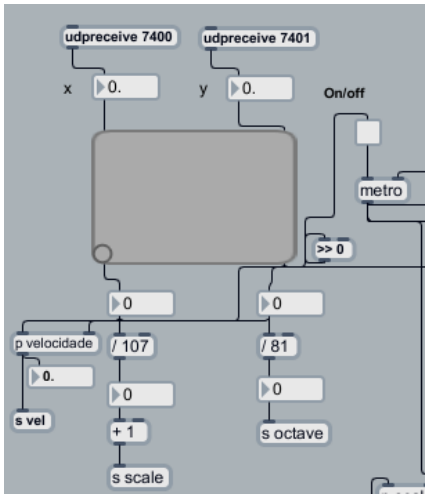


Figure 10. *The beginning of the flow.*

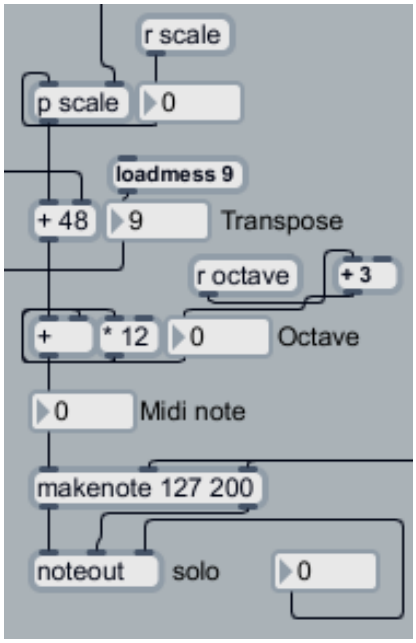


Figure 11. *Where scale and octave are chosen.*

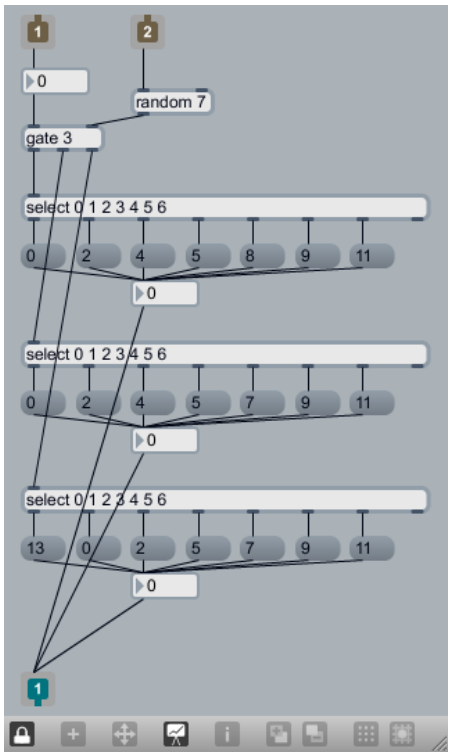


Figure 12. *The scale generator.*

Figure 12 shows the scale generator. It has a *gate* object that receives a number from the X-axis, selecting which *select* object to use. There are three *select* objects with seven outlets each. Each outlet sends a message that corresponds to a note of a scale. These notes are triggered randomly by a *random*, which is receiving information by the “random rhythm generator” (Figure 13).

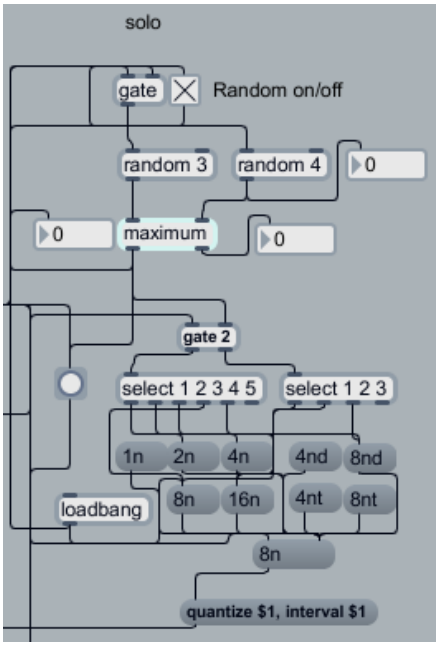


***Figure 13**. Creates a random rhythm.*

The "random rhythm generator" consists of a *metro*[12] that sends *bangs*, triggering messages, which are the rhythmic divisions. The messages are converted into rhythm values, as wholes, eighths, sixteenths, etc. This section also has an interesting object, called *maximum*. This object makes the probability of getting high values bigger than low ones, resulting in an "improvisation" with faster rhythms, making it sound more interesting for the result I wanted.

2.5.2.3 The Rhythmic Section

Inside Live there is one group of tracks called "Drum". This group contains three different tracks that are meant to work as drum parts, in this case the Bass Drum, the Snare, and the Hi Hat. Choosing timbres to work with a juggling routine does not have obvious decisions: circus music goes from the traditional snare rolls to modern electronic music. Glowing balls remind me something modern, related to new technologies, which led me to choose a modern sonority. To achieve this sonority I chose electronic drum

12 The Max help file says that a *metro* "output a bang message at regular intervals".

sounds, instead of acoustic sound samples.

My decision was to use loops for each sub track of the rhythmic section. The ball, depending on its speed, triggers the loops. Also, the timbre of the Bass Drum varies according to the ball's position. This mapping was chosen due a compositional decision, where I relate the intensity of the juggling activity with the drums. So, if the camera tracks the drum's ball, the bass drum loop is triggered, even if it has no movement in the air. If the ball reaches a certain speed, it will trigger the Hi Hat loop. And, if the ball reaches a third speed threshold, it will trigger the Snare loop. Besides, there is a Low Pass filter in the bass drum, which is controlled by the position of the ball in the Y-axis. The higher the ball is, the higher will be the frequencies that will come through the filter.

In this section, I have two groups of information to work with, one that triggers the loops, and other that controls the frequency in the bass drum filter. In the communication between Max and Live, MIDI protocol was used for triggering loops, whereas OSC was used for controlling the filter. A subpatcher calculates the speed of the ball and sends bangs to trigger different MIDI notes. These notes are triggered depending on a speed threshold. Every time a threshold is reached, it is sent a bang to the object "makenote". Then, this object creates a note that is sent to "noteout". A MIDI port sends this data to Live. These notes are MIDI mapped into the loops in Live. When a speed threshold is reached, a MIDI note is sent to Live, triggering the desired loop.

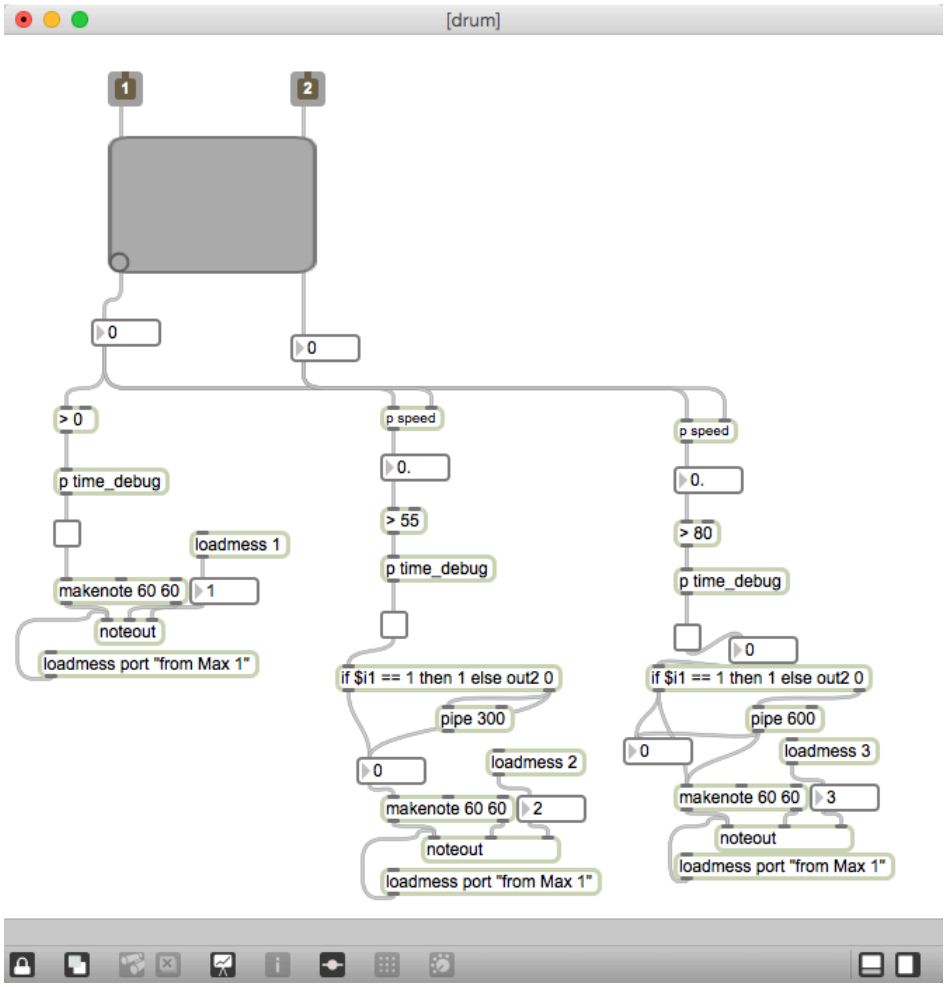


Figure 14. *The subpatcher that sends notes to trigger the drum loops.*

In order to control the filter in the Bass Drum track, at first, I tried to send "MIDI cc"[13] to Live. This method proved to be very unstable, working improperly many times. So, I decided to use another feature from M4L that allows one to "hack" Ableton Live, controlling its API (Application Programming Interface). Their main objects are "live.path" and "live.object". The first requirement for this "hack" is to know the address of the feature to be controlled. Each feature will have a number that corresponds to its address in the Live's API, that can be accessed with the objects mentioned. After setting up my lowpass in the Bass Drum I found its address in the API, or the so-called "path" in M4L language. Then, Max data is sent to the patch inside M4L through OSC. Consequently, this information controls the filter.

[13] "MIDI cc" means MIDI Continuous Controllers, that are used to send continuous information, like volume, modulation wheel, etc.

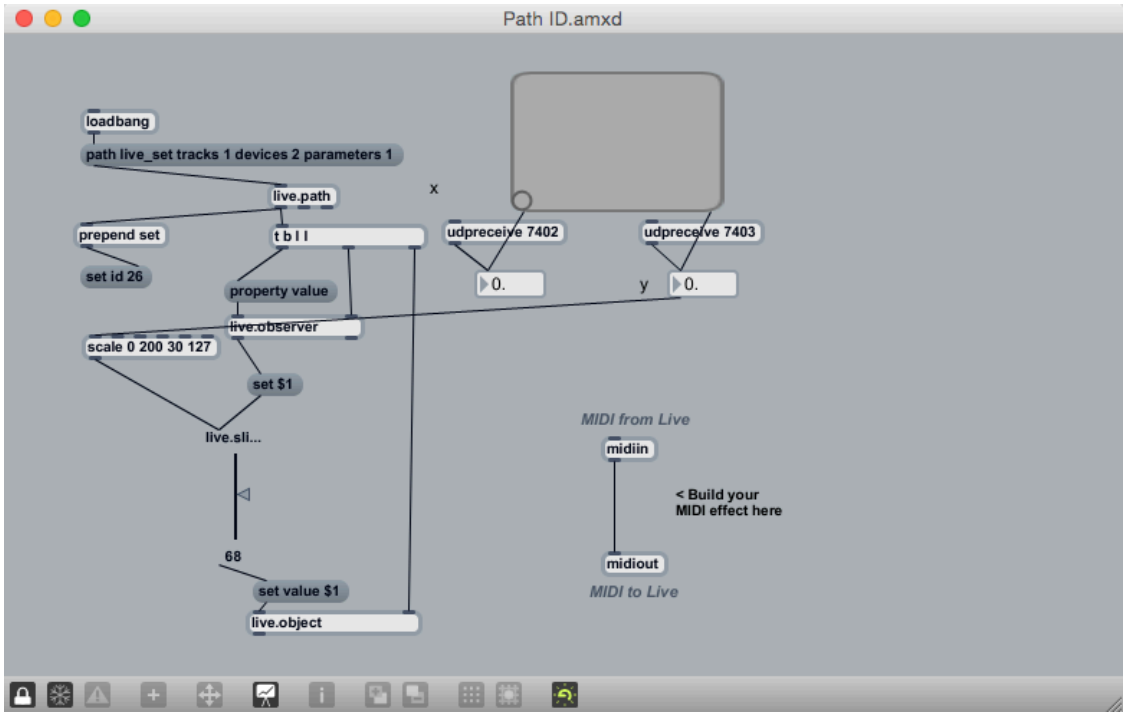


***Figure 15**. The patch that controls the lowpass filter inside the Bass Drum track.*

***Figure 16**. The effects chain inside the Bass Drum track. The first effect is the M4L patch in figure 12.*

2.5.2.4 The Bass Section

Like the drum section, I wanted a modern sound for my bass, which would fit my electronic drum sounds. I wanted the ball to control the timbre, while notes were played automatically each time the bass's ball was tracked by the camera. So, a bass line would be played while the ball was used, and the motion speed would be responsible for controlling the timbre. For the sound design, I was inspired by Ben Burtt's lightsaber sound, in the Star Wars movies. For that, I used a lowpass filter whose threshold frequency is controlled by the ball's speed. The faster the ball moves, the higher are the

frequencies that come through the filter. In summary, a jabbing gesture would result in the sound I was looking for.

To control the filter I used the same procedure as in the Bass Drum, getting its address in Live's API. With that address I used the speed information and scaled it to control the filter inside the virtual synth. The notes were generated in the same way as in the harmony section, with a note generator, that would "improvise" notes in a specific key. The combination of the bass and harmony section gives the feeling of chords.

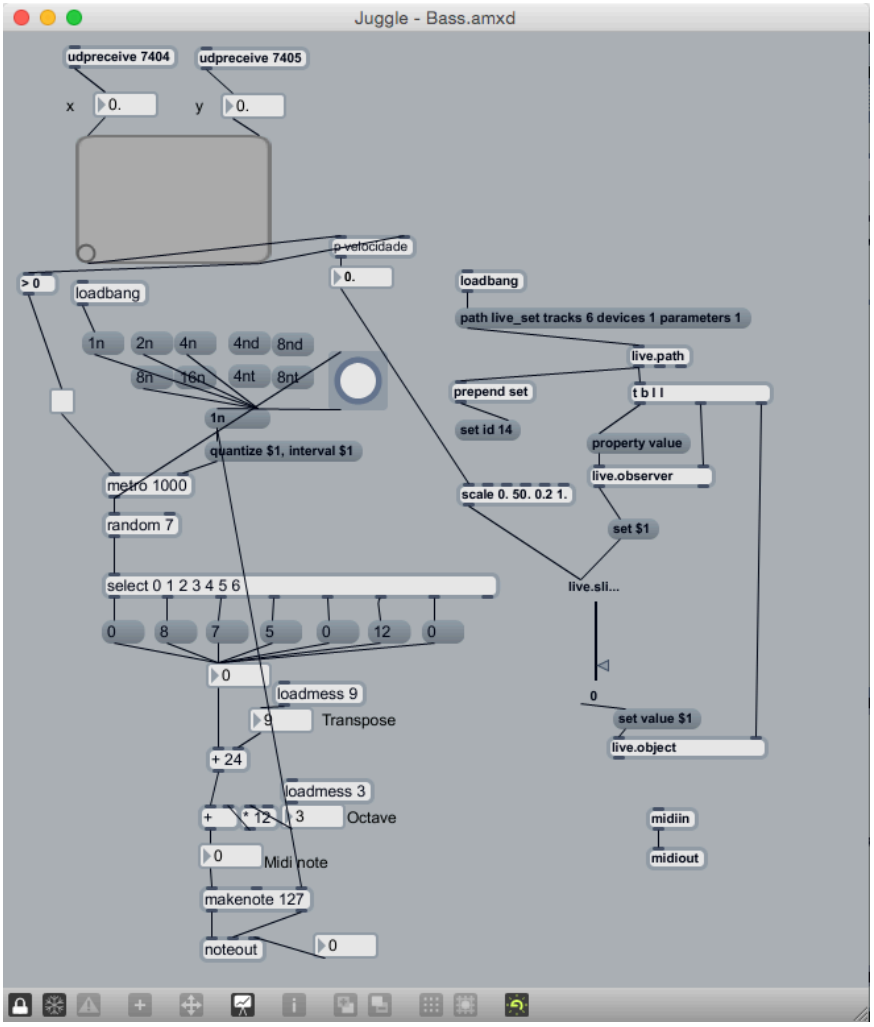


***Figure 17**. The Patch responsible for controlling the bass virtual instrument.*

3 The Performance

The idea of creating this interactive system is allow any juggler/musician to build its own performance, to discover new possibilities and limits of this instrument. As I consider myself more as a musician than as a juggler, my decisions are more musical than visual. So, I decided to gradually build the tension using from the less complex to the most complex sounds. The first section is divided in showing the three balls separately, first the “Theremin synth”, related to the red ball; second, the bass, blue ball; and third, the drums, green ball. Using the first one, I can start performing very slow movements, showing people that the ball is really controlling the sound. Later, I improvise some movements and play with the possibilities. Using the bass one, I start performing some faster movements, and that is how I get the Jedi Lightsaber sound. After some improvisation, I hide this bass one too and get the drum one. Then, using the drum one, I do the same process of showing its possibilities. Next step is getting the blue and the green, having bass and drums together, exploring some throws and how it works to have both in motion, just one in motion, etc. Then, I get the red again, as the Theremin, having three at same time, and improvise with this one in one hand and the other two in another hand. This way, I have to use my ear to accommodate the pitch of the “Theremin” in the harmony and beat created by the other two. Now, it is time to change the mode of the red ball from synth to harmony, that can be made though programming or by another person in the computer. Now, I can improvise my tricks while the music improvises with me. Although I have a plan for the composition, the intuition will be the most important thing in this routine, it is like I was a conductor improvising with my orchestra. Balls will fall, and it is part of the fun, but I know that the music is not going to abandon me.

4 Conclusion

In this work, I brought together different areas that have been appealing to me since a long while: music, programming, and juggling. Creating a controller that is as visual as musical seems to be a good option when talking about watching an electronic music performance.

> In a concert of well-known acoustical instruments, the audience easily understands the relationship between the visual and the auditive - that is, between the physical actions of a performer on an instrument and the sound result thereof -, but in an electronic concert this relationship often remains unclear. Thus, a performance of electronic music is often difficult to comprehend on an analytical level, and, while fascinating to *listen* to, sometimes plainly boring to *look* at. Not without reason, a common complaint about live laptop-players is 'the guy might as well be checking his email'. The underlying cause for this is that the audience might feel puzzled about the *roles* of electronic devices, speakers and persons being present on stage. Even if this puzzlement is totally unconscious, it will still lessen the concentration to the music itself and therefore makes this music harder to appreciate. (WAGENAAR, 2009)

This way, I can offer visual and auditory stimuli simultaneously, similarly to what happens when an instrumentalist plays an acoustic instrument, where the virtuosity of the performer can be appreciated both visually and musically. Besides, it creates the possibility of jugglers that are not musicians to create new kinds of performances, where the juggler is not anymore conducted by the music, but the opposite, he is in charge of it.

This work showed me that the hours spent on juggling, watching tutorials on YouTube of all sorts of things – activities that for a while seemed to be pointless – were essential to construct to what I am now, and to build the knowledge necessary to make this work possible. This is just the beginning, my work with music, technology, and other areas, like juggling, will go on and I will continue following my intuition and passions, doing what I love.

References

BERG, Tamara; CHATTOPADHYAY, Debaleena; SCHEDEL, Margaret; VALLIER, Timothy. 2012. *Interactive Music: Human Motion Initiated Music Generation Using Skeletal Tracking By Kinect.*

DANCEY, Charley. 1995. *The Compendium of Club Juggling*. Bath: Butterfingers.

DRUMMOND, Jon. 2009. *Understanding Interactive Systems.*

WAGENAAR, Arthur. 2009. *Juggling as a controller of electronic music.*

Websites

Cycling74 (MAX/MSP, Jitter)
www.cycling74.com

Juggling Information Service
www.juggling.org

Max Software
http://en.wikipedia.org/wiki/Max_(software)

Open Sound Control
www.opensoundcontrol.org