JÄMME - Kokko

Prototyping an Electroacoustic Musical System with Inexpensive Tools

MUSIC PROCESSING COURSE ESSAY

Juan Ignacio Mendoza
MDP Music, Mind and Technology
April 23, 2013
University of Jyväskylä

1 – Introduction

1.1 - This essay's motivations and goals

In this essay I will give an overview description of the prototype of a musical instrument designed and built using widely available inexpensive technology. Also I will expose some thoughts arisen during the process. I will refer to a software piece that was made as part of a group work in which I was involved, as part of the assignments of the course “Music Processing” of my degree studies at the University of Jyväskylä. However, I will not describe in detail the software coding. Instead I will describe its structure and comment on it.

Therefore this description could be used as a model for building a musical instrument of this kind with other perhaps many different tools, such as computer languages, controller devices and etcetera.

1.2 - Use of Inexpensive Tools

1.2.1 - Ethical Considerations

One special motivation in the project to be described is the use of widely available technological tools, which is coherent with the current philosophical trends in free and open-source software and hardware. This way of proceeding involves the sharing of knowledge to build better products and achieve higher standards than if done otherwise, taking advantage of the communication technologies and the inexpensive hardware and software available.

This way of production is extremely important when thinking about the problems that our world faces, such as the inhuman spread of wealth and damage to the environment. The direct and easy access for everyone to sophisticated knowledge empowers freedom and justice.

Only 30 years ago it would have been impossible to achieve these results in such an easy way, with the equipment used. Now the efforts of many have contributed not only to this but many instances of technological tools that are free and open-sourced.

1.2.2 - Tools of Choice

JÄMME - Kokko was built using the visual programming environment Pure Data, which is open source software, originally developed by Miller S. Puckette and now maintained and developed by a worldwide community. This software can run on the most widespread operating systems and even in old (and thus very inexpensive) computers. Nevertheless, the system to be described was developed mainly using Apple Macintosh computers, from a 2006 G4 to Intel Core Duo machines at the Music Department of the University of Jyväskylä. The audio interfaces used for testing the multichannel were Apogee Ensemble and RME Fireface 800. The speakers used for testing were Genelec 8030-A active studio monitors.

Nevertheless, this system could be implemented using a variety of other hardware since today is relatively easy to acquire o build a cheap multichannel system just by gathering speakers and connecting them to one or more amplifiers. In the same fashion, connections can be done to a computer via one or more cheap audio interfaces, such as the Arduino boards.

2 – Prototyping an Electroacoustic Musical System

2.1 – Kokko: A Musical Starting Point

Normally the design of a musical instrument starts with the need to produce a specific sound or types of sound that would serve to produce a desired musical outcome. JÄMME will be not the exception, but its design can be thought of a means to produce different pieces of music that have some similarities with the one that has served as the starting point for the prototype. The underlying philosophy and theoretical aspects of this process that are here discussed might fulfill a broad range of musical expression intentions.

For the prototyping the sound of fire was chosen just because at an early stage of development of the system I was working on a multi-channel system controlled by a Wii Remote. I was writing the software using an old and slow computer and needed a cost-efficient way to produce an interesting sound which could be moved around a multi-channel system. I found on the internet a Pure Data patch which produces a very convincing sound of fire with very low CPU usage, which will be described further in this text. Then when showing to my colleagues the first version of this software, capable of moving the sound of fire around the performers (we used a 4 speakers array on that opportunity),.one of them thought it as being immersed into the fire. Then a metaphor arose, which we used along the prototyping process:

Many folkloric traditions around the world use fire as a symbol for rituals. For example, in Iran in the new year's they make a small fire and they jump across it to symbolize the burning of the past year's bad things. In Finland they put together a big fire in summer time to frighten bad spirits. They call it "Kokko". Let us imagine that we jump across a big fire together and time freezes when we are in the middle. Fire surrounds us but we do not burn. We can take the flames and start playing with them, discovering all the subtleties of the sound of fire. We play with fire and we literally "play the fire".

2.2 – JÄMME: A Musical Organism

JÄMME stands for *Jyväskylän Älykäs Monikanavainen Musiikin Entiteetti*, which means "Intelligent Multi-channel Music Entity of Jyväskylä". Maybe for this prototype the "Intelligent" word in the name is too ambitious, but that ambition shall be the driving force to keep improving the design after the prototyping stage. Also the "Entity" word of the concept –which fits in the acronym in correct Finnish language- could be better understood with the word "organism", because

a) it is composed by organs and
b) it resembles life.

In the same way he music was described poetically, a simple metaphor can give an overall description of this musical system.

JÄMME can be thought as an organism that remains unborn with a potential for life waiting to be. When born a predetermined life cycle is launched, which is affected in many ways by a stream of inputs which "feed" the system. It as like any living organism would, or example a human being that is conditioned to experience important changes at different stages of life. JÄMME will give an output partly following the conditions of its life and partly reacting to the input.

The major components of this organism (the organs) define the overall design They interact towards a single goal: making beautiful music. Now, what should be "beautiful music"? That is what this organism's creator (me) has decided in the design of the music, which it is comprised by the complex set of rules that determine the birth, life and death of it. In the following sub-subsection a more detailed explanation of these "organs" is given.

2.2.1 - Synthesis.

The core synthesizer module produces three currents of raw fire sound, which are then processed by four processing units: Resonant Filters, Pitch Shifters, Cutters and Granulator. Another module mixes all the currents before they are spatialised.

The output of each processing unit could be linked to the input of any but currently this is not implemented and the interconenctions are rather simple, following mainly the order Raw Fire, Resonant Filters, Pitch Shifters and Cutters. The granulator input is directly connected to the Raw Fire output.

2.2.1.1 – Raw Fire.

The core synthesizer module for fire was mainly built upon the code by James Farnell in a patch retrieved from his website. It consists of three main elements:

- Lapping: A low frequencies rumbling noise, resembling the sound of a vigorous flame.
- Crackling: A mid-high frequencies sudden burst, resembling the sound of the burnt material breaking.
- Hissing: High frequencies short burs, resembling the sudden emanation of gases.

This main unit produce three flows of digital audio, each one is a sound of raw fire similar yet distinct from one another. Each of these three flows can be thought of as a “stream of sound.

2.2.1.1 – Resonant Filters.

There are three band-pass filters set with a quite narrow bandwidth, by default filters 2 and 3 are set to produce intervals of a fifth and an octave respectively, from the first filter. Again, each of these outputs can be thought of as a stream of resonant filtered fire sound.

2.2.1.2 – Pitch Shifters

There are three pitch shifters built upon code by Hans Roels. Each unit has a continuous range of two octaves.

2.2.1.3 – Cutters

These are three modules that slice each audio current. The time of slicing can vary from 20 to 1000 milliseconds (metronome tempo from 300 to 60). A random component has been incorporated to make subtle variations of tempo, so that it sounds less artificial.

2.2.1.4 – Granulator

The delay line granulator was built upon the code by Hardoff. Currently receives input directly from the first Raw Fire stream. The parameters used are pitch, duration, delay, metronome speed, filter cutoff, auto-panning, delay feedback and delay feedback time.

2.2.1.5 – Mixer

The final stage of the synthesis is a module that mixes the outputs of the three streams of Raw Fire, Resonant Filters, Pitch Shifters, Cutters and the two outputs of the Granulator to then be distributed to the speakers by the spatialisation module. Additionally it has a cross-fader that allows balancing between an overall output of raw fire and processed sound.

2.2.2 – Spatialisation

This module has two sub-modules, ecah one distributing different streams as explained below. Ecah sub-module also, uses the VBAP panning technique for spherical speakers arrays (Pulkki, 1997). Therefore the input of the spatialisation sub-modules is polar coordinates. However, JÄMME only uses two dimensions (a flat circle), therefore special coding had to be written for the conversion to polar coordinates from the Cartesian coordinates that are input, for example by the Wiimote IR, as explained further in section 2.2.3.

Testing was made mainly with two and four main output channels. When using the latter, the speakers were placed in a square around the performers. However the system is designed to work with a concert eight-speaker circle surrounding the audience and the performers, it could adopt other speaker setup.

2.2.2.1 – Three-stream spatialisation

This sub-module distributes the three distinct streams emanated from the mixes of Raw Fire, Resonant Filters, Pitch Shifters and Cutters. They can be positioned freely at any point of the area of the circle.

2.2.2.2 – Stereo spatialisation

This sub-module distributes the two streams emanated from the Granulator. Each of these streams can move over the perimeter of one half of the circle, left or right.

2.2.3 - Wiimote IR

A Nintendo Wii Remote game controller was used as the input interface. Only the Infrared (IR) camera was taken advantage of. Working along with Darwiin Remote OSC it tracks the existence/non-existence and position in a Cartesian plane of IR light sources. In this case three candles were used as light sources, to be coherent with the poetics of the music. Darwiin Remote OSC sends the position of the candles to Pure Data by means of the OSC protocol through an internal network port.

2.2.4 – Life Cycles

2.2.4.1 – Main algorithm: The ADN

A Fibonacci series algorithm is used as the main proportions generator. It has 10 levels of depth and is activated by a metronome counting in steps of 1/100 seconds. At every step the system counts up to the next number in the series and generates a new series. This process is done up to ten levels, generating a micro and macro structure. The greater "concert" structure is comprised of 4 time segments strating at 0, 90, 234 and 378 seconds. These major temporal structures are called "life cycles", which have a predetermined internal behaviour that can be affected by the input.

2.2.4.2 – Life cycles: Major Musical Sections

Each life cycle has a particular behaviour that controls parameters in each of the synthesis processing modules, as well as other parameters in the entire organism. For example, the first cycle starts with raw fire and activates switches that start each raw fire synthesizer when a new point is detected by the IR camera. This means that the system will be in stand-by until a candle is lit and only then a sound will be heard. When the other two candles are lit the other two raw fire synthesizers will start. Also in this first stage each candle controls the spatialisation of its current. Gradually the mixer's crossfader is moved from the raw fire position to the processed position as the faders for resonant filters and pitch shifters are raised, at different ratios. At some point

also the fader of the cutter is raised. At round 90 seconds the system will progress to the second cycle, where another distinct behaviour will develop.

3 – A further implementation: Artificial Intelligence

When this text was completed JÄMME was still in an early stage of development and Artificial Intelligence was not yet implemented. Nevertheless some outlines for this implementation have been done, which are here commented.

First, let us consider that our desired musical intelligent agent would act as a performer, a machine able to create and improvise music, learning from some kind of music, thus developing its "sensibility" by means of a cognitive structure whose goal is to keep interaction with the input in a musical way, based on the widespread assumptions that define musical structure: repetition and variation.

A very simple way to achieve this could be the implementation of a basic learning schema in which the system tracks the input and stores certain characteristics of it (e.g. amount of movement, distance, acceleration), then gives back an output majorly affected by this data, then "listens" for a new input, compares it with the previous input and output and makes a decision of what to do next: vary or repeat. This decision could be made by simply following a predetermined conditional statement (e.g. "if, then, else"). For example, "if time is X then take decision Y, else proceed as Z", which is extremely easy to code. Time patterns could be given by a more complex yet easy to implement algorithms, such a as the aforementioned Fibonacci series generator. Then, the process will continue interacting with the input.

JÄMME – Kokko is available at http://juanignaciomendoza.weebly.com/mmt-resources.html.

5 - References

5.1 – Bibliographical reference

Pulkki, V. Virtual sound source positioning using vector base amplitude panning. Journal of the Audio Engineering Society, 45(6) pp. 456-466, June 1997.

5.2 - Third-Party Software

A list of software used in its original form, with minor changes or just worth mentioning:

Pure Data Extended v0.42.5 and 0.43 by Miller Puckette and Pd community. Retrieved from puredata.info (website)

Three-component fire synthesizer (Pure Data patch) by James Farnell. Retrieved from http://obiwannabe.co.uk/.

Simple-Pitchshift (Pure Data patch) by Hans Roels. In the collection Abunch049. Retrieved from http://home.base.be/hanstine/hans/index-eng.htm

Grannie Basher (Pure Data patch) by "Hardoff" (username, real name not available). Retrieved from http://puredata.hurleur.com/sujet-1932-grannie-basher-delay-line-granulator

Darwiin Remote OSC 0.3.2 by Andreas Schlegel based on the original Darwiin Remote by Hiroaki Kimura. Retrieved from http://code.google.com/p/darwiinosc/