Title
Transforming the teponaztli

Author
Garcia-Munoz, Issac

Publication Date
2013

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA, SAN DIEGO

Transforming the Teponaztli

A Thesis submitted in partial satisfaction of the
requirements for the degree
Master of Arts

in

Music

by

Issac Garcia-Munoz

Committee in charge:

Professor Tamara Smyth, Chair
Professor Miller Puckette
Professor Richard F. Moore

2013

The Thesis of Issac Garcia-Munoz is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

____

____

____
Chair

University of California, San Diego

2013

DEDICATION

To a family out of sight, but never out of mind.

EPIGRAPH

I wanted my children to see examples of real Mexican heroes. I grew up thinking that Mexicans could only wash dishes and work in fields.

–Victor Villaseñor

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to thank the members of my committee for their support. I would also like to thank Marcelo Lazcano Flores for his collaboration in writing and performing *Dos Mil Doce*, the UCI-UCSD telematics group, and Ryan Nestor for playing the teponaztli. And I would like to thank the UCSD Music Department for letting an engineer play in the musician's sandbox.

VITA

2007	B. S. in Electrical Engineering, California Institute of Technology, Pasadena
2008-2011	Test Engineer, Broadcom, Irvine
2011-2012	Graduate Teaching Assistant, University of California, San Diego
2013	M. A. in Computer Music, University of California, San Diego

PUBLICATIONS

Issac Garcia-Munoz, "Enhancing Silicon", *Caltech Undergraduate Research Journal*, volume, 2004.

ABSTRACT OF THE THESIS

Transforming the Teponaztli

by

Issac Garcia-Munoz

Master of Arts in Music

University of California, San Diego, 2013

Professor Tamara Smyth, Chair

Transforming the teponaztli is a renovation of the Aztec log drum in which a new electroacoustic instrument is formed. Two prototypes were constructed to arrive at a design which had the desired balance of acoustic sound and electronic signal. Applying signal processing effects to the teponazlti was first attempted for the piece *Dos Mil Doce*. Then the issue of the percussionist controling the effects without taking the focus away from the performance was addressed. Trials to extend the control in order to push it towards the domain of shared/multi-user

instruments were carried out. Methods for capturing different percussive strikes and gestures and using those to control the parameters were more successful. These were implemented using signal analysis and infrared sensors.

Chapter 1

Introduction

1.1 History and Motivation

In the summer of 2012 I visited the Los Angeles County Museum of Art's exhibition *Children of the Plumed Serpent: The Legacy of Quetzalcoatl in Ancient Mexico* where I first saw a teponaztli which was behind a glass showcase. There was only a single sentence describing the drum as one which was to be played with mallets during celebrations. Details of this particular type of instrument have been preserved on stone monuments in the shape of teponaztlis which honor Xochipilli, the god of art, dance and song [Par05].

The exhibit reminded me of an interview I conducted with Augustine Mamani, an Andean flute builder in El Alto, Bolivia. He wanted to start a museum with interactive exhibits that show how and when the Andean instruments should be played. I was inspired to revive the teponaztli in my own way and transform it

Figure 1.1: Stone teponaztli [top], wood teponastli [bottom]. By Madman2001 (Own work), via Wikimedia Commons.

to reflect its patrimony and the present technological progress.

1.2 Computer-Extended Acoustic Instruments

The ability to process the signal from the two tongues places the electroacoustic teponaztli in the category of computer-extended acoustic instruments. Using the computer to add layers of effects to the acoustic sound opens up a wide range of possibilities, but there must be a choice to make when considering what processing to implement. My initial goal was to augment the sonorous pallete of the instrument. And I didn't want the result to be just a drum pad in the shape of a teponaztli. When creating the computer-extended acoustic instrument, the extensions would be done primarily by post-processing the acoustic signal rather than by using additional sensors. The idea was to minimize the interference that any sensors or electronics would have on the user's ability to play the instrument.

The Wildlife ensemble [SJ93] and SABRe clarinet [SS] are examples of computer-extended acoustic instruments which inspired the development of the electroacoustic teponaztli. Wildlife, an ensemble involving an electric violin with a Zeta pickup and the Mathews/Boie Radio Drum which share the same pitch space so the output from one instrument, say a glissando from the violin, can change the pitch of the drum [SJ93]. The Zeta pickup has separate piezoelectric sensors under each string of the violin and a module to convert polyphonic pitch to MIDI. The Mathews/Boi Radio Drum uses RF transmitters on the drumsticks and an antenna array on the drum pad to capture three dimensional position of the sticks[1]. The computer forges the relationship of the improvisational ensemble's sound output and makes the two instruments interdependent. The ability of one performer to surprise the other breeds new and interesting performances which may not have occurred otherwise.

The Sensor Augmented Bass clarinet Research (SABRe) project at the Institute for Computer Music and Sound Technology at Zurich University of the Arts has developed a clarinet with embedded sensors whose data is streamed wirelessly using Open Sound Control (OSC) [SS]. This is a seamless integration of sensors

[1]http://www.mistic.ece.uvic.ca/research/controllers/radiodrum/

which handle the note, movement and pressure tracking without inhibiting the instrument's acoustic qualities. Chapter 4 shows how drum strike information was obtained for the teponaztli using signal analysis instead of using sensors like the SABRe. This was done to minimize the number of electronic components needed on the instrument.

1.3 Electronically Shared Instruments

In the early stages of development of the electroacoustic teponaztli, an objective that arose was to use the computer to distribute the control of the instrument to multiple users both locally and telematically. Networked instruments date back to Perkis who led the Hub, a group of computer musicians in the 1980-1990s, which can be thought of as one larger computer instrument [BP00]. Another example of sharing control is the group D'CuCKOO which uses technology to include audience participation in their performances by having a large inflatable MidiBall in the audience that sends MIDI data wirelessly to trigger new sounds and visualizations with each bounce[2].

Even more closely related is the Tooka, a two person wind instrument with various sensors that are fed to Pure Data (Pd[3]), the real time graphical programming environment for audio, video and graphical processing[4] [FKTM04]. Air pressure controls dynamics, buttons control pitch and bending the Tooka causes pitch

[2]http://www.telecircus.com/yeold/Side/Dcuckoo/
[3]Pd is also the program used to process the teponaztli in live performance.
[4]http://puredata.info/

bending. This is a very intimate performance scenario which should contribute positively to the expressivity of the instrument. The mapping of the buttons was setup in such a way as to promote coordination between the two players which is a restriction analogous to sharing an acoustic instrument. Players can catch and release notes and add layers of drone textures. It seems to be an excellent exercise in communication and coordination between the performers through a controller for a computer instrument. Accelerometers and torque sensors are planned additions to this instrument. Unlike in this work, the aim of the Tooka was not to use the instrument's acoustic signal for control but the idea of a multi-user instrument is strongly related.

1.4 Transformation of the Teponaztli

The construction of the first prototype of the electroacoustic teponaztli is described in detail in Chapter 2. I refer to this instrument as a prototype because after being played in the piece *Dos Mil Doce*, described in Section 2.2, its shortcomings led to the construction of the current version which is described in Section 3.3. Measurements on a simplified PVC prototype model described in Chapter 3 informed the design of the current version. After efforts to create a shared instrument and distribute the control over the effects parameters, as shown in Chapter 2, methods for using signal analysis to extract the parameter values from the percussionist's performance is the topic of Chapter 4. The result is a computer-extended electroacoustic teponaztli which I hope revives the spirit that the Aztecs instilled in the instrument and adds it to the modern computer music tradition.

Chapter 2

Stages of the New Teponaztli

2.1 Construction of the First Prototype

The construction of the electroacoustic teponaztli began with a tree trunk section about 6” in diameter and 18” long. The piece had been discarded by city maintenance crews in routine tree maintenance making the construction of the instrument a sustainable process. When starting to carve the wood, the only reference I had was the teponaztli from the museum exhibit and the instrument building experience I gained as a Watson Fellow[1]. Armed with a saw, chisel and mallet I decided to cut off four inches from each end and hollow out the center piece where the tongues were to be cut out and have this be the resonant cavity (See Figure 2.1). I was familiar with this method from making a djembe and chose

[1] I traveled through Latin America interviewing instrument builders and learned to make the following instruments: an acoustic guitar in Buenos Aires, Argentina, bombo leguero in Santiago del Estero, Argentina and a djembe in Santiago, Chile.

it also because I thought it would be the fastest way to hand carve the cavity. Traditionally, the teponaztli is carved out directly under the tongues instead of being hollowed out through the center of the trunk.

Figure 2.1: Boring through the eucalyptus tree trunk.

In designing the dimensions of the tongues and specifying the tuning I decided to hold the thickness and widths constant and change the length. In finding the harmonic structure of the tongues I used a one-dimensional approximation with a node where the tongue is joined with the rest of the trunk and an anti-node in the middle where the tongue is free to move. In Section 3.1 I will show the investigation in which the vibration of the tongues are calculated using the solution to the three-dimensional equation of motion for a bar fixed at one end. In the one-dimensional model one quarter wavelength of the fundamental frequency fits in the length of the tongue (See Figure 2.2). I cut the first tongue out to have a length of 14 cm so the fundamental for low pitched tongue should be $f_1 \approx 607Hz$, where the frequency $f = c/\lambda$, where c is speed of sound (approximately 340m/s), and λ is the wavelength (4×14cm for this tongue). Therefore, to get the required length of the other tongue in order to achieve a major fifth interval (a personal choice that is not necessarily traditional), I needed to satisfy the 1.5 ratio so that $f_2 = 1.5 \times 607 = 910.5Hz$ and the length of the second tongue became $340m/s/4 \times 910.5Hz = 0.0933$m or 9.3 cm.

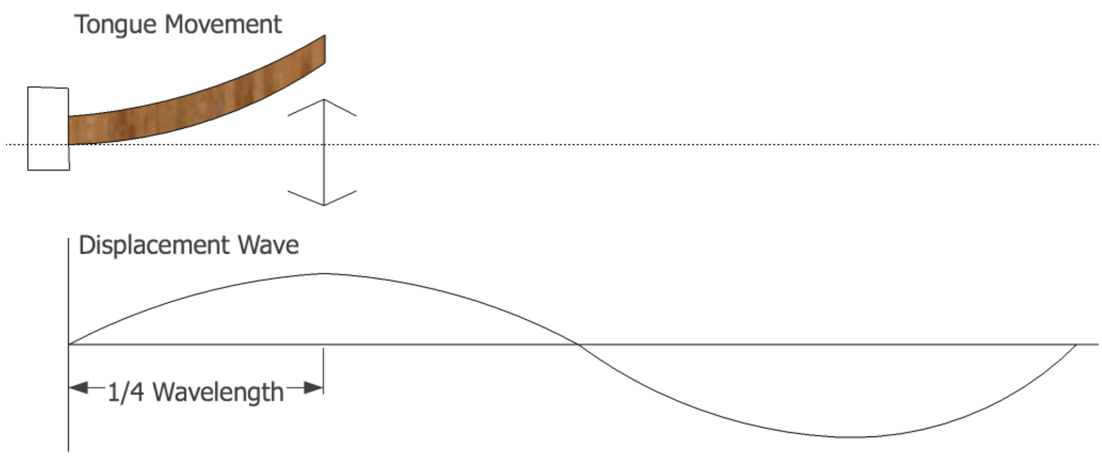

Figure 2.2: One-dimensional model of the tongue showing the tongue movement and the length of the tongue is one-quarter wavelength of the fundamental displacement wave.

Once the dimensions were established, two piezo contact microphones were glued to the underside of each tongue. I was worried about loosing the low frequencies due to impedance mismatch so I connected audio transformers between the piezos and the 1/4" female jacks that were mounted on the side of the teponaztli in order to bring down the impedance that the preamp sees. Another option would have been to build a buffer circuit in addition to the transformers, but I chose to forego this addition to minimize the number of components that reside inside the instrument.

At this point in the construction process I reattached the end pieces and recorded the output of the piezos. I found that the amplitude of the response signal decayed very rapidly. I had hoped to get more resonance from the tongues, or a more sustained fundamental. I then tried two modifications to improve the acoustic sound of the instrument to fill in what the electric signal was missing. First I cut holes in the center of the sides which were smaller in diameter than the diameter of the inside cavity in hopes that there would be a Helmholtz resonance effect. The problem with this was that both sides were opened and the dimensions were not designed with the Helmholtz equation in mind. The other change was trimming the thickness of each tongue so that they would be able to vibrate for a longer period of time with less mass. Each one of these modifications was an improvement, but the balance was still not satisfactory so I was motivated to make a second version of the instrument discussed in Chapter 3. Performing with the instrument was an important step before designing the next version of the teponaztli.

2.2 Initial Performance

The first prototype of the electroacoustic teponaztli was finished in time for the December 2012 UCSD Computer Music Concert held at the Experimental

Theater in the Conrad Prebys Music Center (CPMC). The piece performed is titled *Dos Mil Doce* (Two Thousand Twelve) and was written in collaboration with Marcelo Lazcano Flores, a colleague from the composition department. The idea was to exaggerate the ridiculous end-of-the-world theories that arose from the end of the Mayan calendar. The fact that the teponaztli is an Aztec and not a Mayan instrument added to the satire. In the following two sections I describe the processing that was performed on the instruments for *Dos Mil Doce*.

2.2.1 Instrument Processing

Now that the teponaztli was transformed into an electroacoustic instrument, I was able to analyze and process the sound from the instrument. For this piece in particular I chose to use Pd for the processing and the Pd external +bubbler~[2], a granular synthesis delay. Eight parameters were MIDI mapped to be controlled by rotary knobs on an M-Audio Oxygen8. These were delay volume, delay time, time variation, percent feedback, resonance, grain size and grain reversal. I incorporated this type of delay processing so that the teponaztli could add texture in the piece. The only rhythmic line for the drum is a simple pulse to mimic the tick tock of a clock and rotate a projected image of an Aztec calendar on each tick. This was done with the Graphics Environment for Multimedia (GEM) in Pd by sending the audio output of the teponaztli through an envelope follower, which caught each attack, and then to the GEM object which rotated the image clockwise on each

[2] +bubbler~was written by Tom Erbe and is part of the SoundHack DSP suite: http://www.soundhack.com/freeware/

strike. A simple looper was also created and mapped to the play, record, stop buttons on the Oxygen8. I included this to avoid having silence whenever the parameters were changed.

I wanted another effect, in addition to the granular delay, that would make the teponaztli interdependent with another instrument in a manner similar to the pitch space sharing of the Wildlife ensemble mentioned previously in Chapter 1. The process of convolving the output of the teponaztli with that of another instrument was used as an effect that fuses the two instrument's frequencies reinforcing their common modes and rejecting their differences. The interaction in this case is not a clearly defined pitch mapping like that of the Wildlife ensemble, where each performer could change the other's output, but a layer of their recombined signals which could be added to the unprocessed sound. For *Dos Mil Doce* the other signal came from an electric guitar played by Marcelo. The fact that the signal from the piezo pickups on the tongues of the teponaztli had such a short decay caused the convolution process to behave unexpectedly. The convolution turned out to be more like a gate effect in which there was only output when both instruments were playing and the drum was the trigger which passed the guitar mostly unchanged. The frequency response of the piezo pickups from a drumstick strike on the tongues can be seen in Figure 2.3. It shows that there are no dominant resonances and the energy is distributed fairly evenly amongst all the frequencies. When convolving the guitar with a frequency response which is distributed in this manner there are less frequencies to reject and therefore the guitar passes mostly unchanged. We used this to our advantage by writing in a part in the piece where the guitar plays arpeggios which are inaudible until hits from the teponaztli let some notes through and surprise the audience.

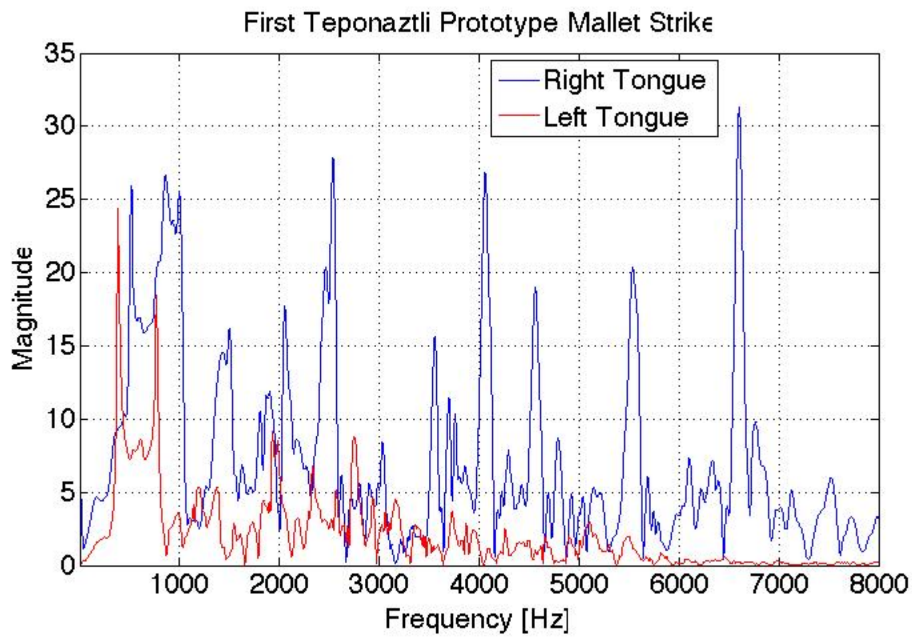

Figure 2.3: Frequency response of the first teponaztli prototype

The implementation of the convolution was performed in the frequency domain with a four time overlap using the Hann window and an FFT block size of 1024 samples. When running an FFT to analyze a single note, as is done for all the frequency response figures in this work, the waveform is zero padded in the time domain (has silence before and after the note) so the use of a window function is unnecessary. For live performance, however, the overlap and windowing are necessary because the input to the FFT could be a long signal (drum roll for example) and not an isolated note. For more detail regarding the overlap-add and windowing for this fast convolution method I refer the reader to Section 2.4.8 of Moore's Elements of Computer Music [Moo90]. The FFT convolution in Pd takes the signals from instrument A and instrument B and sends them into rfft objects which output the real and imaginary values of a fast Fourier transform, notated as the real R_a, R_b, and imaginary I_a, I_b. Then a complex multiplication is performed which corresponds to convolution in the time domain.

$$(R_a + jI_a)(R_b + jI_b) = R_a \times R_b + jR_aI_b + jR_bI_a - I_aI_b$$

The real $R_a \times R_b - I_aI_b$ and imaginary $R_aI_b + R_bI_a$ results of the multiplication are then sent to rifft object which computes the inverse Fourier transform with those inputs [Puc07].

2.2.2 Vocal Processing

I felt it was important for the narrative of *Dos Mil Doce* to match the sound processing applied to the computer-extended teponaztli and I go into detail in this section as to how that was accomplished. I explored the use of algorithmic composition to perform a musical recombination of an Aztec poem. The python based program, AthenaCL, written by Christopher Ariza, was used to do the recombination. It was inspired by Ariza's bablecast, the "algorithmic, computer-generated podcast series created from fragmented and distorted sounds of U.S. and World leaders"[3].

[3]http://www.flexatone.org/article/babel

Whereas bablecast was built exclusively from sounds harvested within a defined period of days the narrative in *Dos Mil Doce* is made using a Nahuatl[4] text, *Thirst for Immortality*, and its translation into Spanish *Sed de Inmortalidad*:

Niwinti, nichoka, niknotlamati,
nikmati, nik-itoa, nik-elnamiki:
maka aik nimiki, maka aik nipoliwi!
Inkan ahmicowa, inkan ontepetiwa,
in ma onkan niauh:
maka aik nimiki, maka aik nipoliwi!
(Nahuatl)

Me siento fuera de sentido,
lloro, me aflijo y pienso,
digo y recuerdo:
Oh, si nunca yo muriera,
si nunca desapareciera!. . .
Vaya yo donde no hay muerte,
donde se alcanza victoria!
Oh, si nunca yo muriera,
si nunca desapareciera. . .
(Spanish)

I wanted the text to play with increasing layering of the spoken word, both in Nahuatl and Spanish to represent the supposed end of the Mayan calendar on December 21, 2012. The piece has a correlation with the passage of time starting with the tick tock played by the teponaztli and then the introduction of the guitar which shows the Spanish influence on both the Mayan and Aztec people. The choice of the Nahuatl poem, even thought it is not a language of the Mayan people, was made to satirically promote the recent obsession with the supposed Mayan predictions of the end of the world. In *Sed de Inmortalidad* the poet ponders what it would be like never to disappear and to reach victory by

[4]Nahuatl is the language of the Aztecs who lived in what is now central Mexico.

moving to a place without death. The narrative remix is the soundscape over which the electroacoustic teponaztli and guitar play to sound the finale to *Dos Mil Doce*.

Csound instruments were used in AthenaCL to achieve the desired text reconstruction. I started with SampleUnitEnvelope, Csound instrument #32. When using the default settings only the first second in the clips were played so in order to hear whole words the rhythm was changed to quarter notes. The auxiliary parameters x0-x6 deal with the sample playback. I changed x5, the audio file start time, so that I could get more than just the first word. I was able to randomize the start time within the clip which was only about 30 seconds long. I noticed that I could introduce pauses or silence if I specified a range for the random start time that was out of the file length. Since this was an unwanted effect I made sure the range was within the file length. Next I wanted to try layering words by creating multiple copies of the same audio file and changing an auxiliary parameter to search entire directories. I found that the better way to layer is just to make copies of the texture instance using ticp. At this point all the instances played at the quarter note rhythm so the fade-in/fade-out was very noticeable. The AthenaCL commands used can be found in Apendix A.

I found the desired density, but I wanted the remix to gradually progress from being coherent to unintelligible. I thought that by changing the rhythm parameter from just a quarter note loop to a genetic algorithm (using gaRhythm) the rhythm could mutate from quarter notes to shorter values. I found that a short, three note string is not long enough for the rhythm to change gradually. The gaRhythm by itself was not creating the gradual progression I sought out so then I tried changing the sample start parameter from random to a Fibonacci series. Instead of using the fibonacciSeries object I went with the generator parameter object cyclicGen for the sample start times. The clip would be sampled from 0 to 20 seconds in 0.5 second intervals. I also wanted the Nahuatl text to be heard first and then after 20 seconds for the Spanish translation to be heard so I changed the texture instance time parameters accordingly.

The resampling ran two minutes long with the second half adding more and more layers with different parameters so that it became unintelligible and chaotic. For one texture instance I used the iterativeRhythm parameter objects to have a disrupted loop. In another I used the gaRhythm mentioned previously and then another with the sampleReverb Csound instrument #30. The process to shape the final form took a lot of trial and error (see Apendix A). The limitation of AthenaCL is the difficulty to specify the overall form of a piece because the texture instance format leads to an additive process where one builds up using texture instances from start to finish. This difficulty, however, may be more obtrusive when one has a clearly defined idea of what the output should be. If the composer does not have a strict form in mind for the outcome then less tweaking of the parameters is necessary. The great benefit of using the Csound sampling instruments from within AthenaCL is that once a form is created, it can be saved as an xml file that can be later recalled and used with new audio clips. It may have taken less time to generate the resampling by cutting and pasting in a digital audio workstation, but having the AthenaCL xml file affords fast remixing with new texts.

An improvement to this process would be to automatically determine the start times of each word instead of guessing that a quarter note value is enough to hear at least one word. This could be accomplished by using a speech recognition toolkit, such as Sphinx[5], to make a list of the times in the audio file where each word is spoken. Then this list could be implemented in the sample start time parameter of instrument #32 using basketSelect so that the samples would be triggered precisely at the start of each word.

Using the Sphinx toolkit I was able to create a program in C that would output the time at which the speech recognition function found a new word. There is a speech model that has to be loaded, where the default model is a simple English library. If I wanted to use this on Nahuatl I would have to create my own speech model that Sphinx would then load. An audio file with the phrase go forward ten meters was used to test the program. The code for the Sphinx program and its use can be found in Apendix A.

2.3 Telematics

An opportunity to experiment with the teponazlti as a networked instrument came with a UCSD Telematics[6] course set up by Professors Mark Dresser

[5]http://cmusphinx.sourceforge.net/

[6]Telematics defined here as the distributed multi-media performance with integrated auditory and visual immersion.

and Roger Reynolds. Convolution processing extends the interconnectivity of the instrument to include another signal so that the combined sound output is dependent on all the instruments. The output of the teponaztli before being convolved was solely dependent on the percussionist. The idea to try out was to have the control of the granular delay be removed from the percussionist. In theory there is no limit to the distance between performers while using current technologies like JackTrip and OSC. In addition, the use of a computer makes the mapping between gestures and instrument output arbitrary. Any movement can be mapped to a sound or parameter. This eliminates the physical restrictions on the construction of the instrument that were in place to make it acoustically sound. But there needs to be a balance between simplicity and utility [BF03].

The course was a partnership with UCI with weekly telematic rehearsals. My group included Marcelo Lazcano Flores (electric guitar) and myself at UCSD and Michael Mathews (synth) and Jonathan Mattson (drums) at UCI. Skype was used for the video projection with a stationary camera at each location showing each ensemble and JackTrip for the audio connection. For documentation purposes there was a roaming camera to focus the attention on a particular instrument and create a more cinematic experience. In post production this would be coupled with corresponding audio mixing and panning. The target audience would be live online or streamed at a later date. During the course there were only rehearsals, however, for a real performance I proposed using LiveStream technology as a relatively inexpensive broadcast solution to mix the four video feeds and six audio channels.

I worked with Michael so that he would be able to control the delay parameters from his Korg synthesizer through the network. This was done using UDP (User Datagram Protocol) because it was only control data so a few dropped packets are preferable than higher latency. The intention was to see how the percussionist would react to someone else changing the sound output of the instrument without knowing what processing is taking place. In choosing the constraints or limits for the parameters I had hoped that the changes would not seem like an arbitrary remapping of the processing, but a smooth transition. Another experiment that I would have liked to run was to send the processed signal from the UCSD group to the UCI group but only monitor the clean signal at UCSD. In an improvisation setting this would amount to a lossy feedback loop since the UCSD group would not have all the acoustic information that the UCI group is reacting to. I believed that this would free the teponaztli percussionist from following the actions of the controller yet the multidimensionallity would still be present in the group as a whole. Unfortunately the quarter ended before we were able to incorporate this experiment into the ensemble.

Even thought we did not have time to extend the control of the teponaztli across the network to the other campus, the drum was played as a locally shared instrument. Ryan Nestor, a percussionist, played the teponaztli while I controlled the Oxygen8. When I increased the amount of feedback and delay time to create a denser texture Ryan refrained from playing long sequences of short strikes which was what I had expected. When I eliminated the feedback, reduced the dry signal and increased the delay time, there was a pause in his playing which I believe was to recalibrate his technique for the new delay time. What I learned from a professional percussionist such as Ryan playing my instrument was his exploration of the entire instrument, not just the two tongues. This lead to the idea of moving the control of the parameters back to the percussionist, but in such a manner that they change based on the percussionist's technique and/or gestures. This is explored further in Chapter 4.

Chapter 3

Development of the Latest Teponaztli Using Acoustic Measurement and Analysis

The unsatisfactory acoustic and electric signal resonance of the first prototype led to the creation of a PVC prototype model of the teponaztli in order to explore the use of electret microphones instead of piezo pickups, improve the acoustic resonance, find the best location to mount the microphones and compare the measurements of the model with theoretical calculations. Differentiation of the strike location, discussed in Chapter 4, is another goal that I believed could be achieved through the analysis of the signal. The first step was to characterize the acoustical properties of the PVC prototype model because it could be easily built and modified. Before doing so I describe the vibrations of a bar held at one end (a model of the teponaztli's tongues) in more detail.

3.1 Vibrations of a Bar Held at One End

In Section 2.1 I show how I tried tunning the tongues by calculating their fundamental frequency using a one-dimensional approximation. Here I explore the use of a more accurate approximation to see if it would yield results closer to those gathered from the frequency response of the teponaztli. Rossing presents the solution to the transverse displacement for a bar held at one end [FR98] where the frequencies of the nth partial are:

$$f_n = \frac{\pi K}{8L^2}\sqrt{\frac{E}{\rho}}[1.194^2, 2.988^2, 5^2, \cdots, (2n-1)^2] \tag{3.1}$$

where K is the radius of gyration, $K = \frac{H}{\sqrt{12}}$, H is the height of the bar, E is Young's modulous, ρ is density, and L is length of the bar (See Figure 3.1). This solution is derived by applying the boundary conditions for the bar clamped at one end:

$$y(x)|_{x=0} = 0, \frac{\partial y(x)}{\partial x}|_{x=0} = 0$$

to the general fourth order solution for transverse waves in a bar:

$$y = \cos(\omega t + \phi)[A\cosh kx + B\sinh kx + C\cos kx + D\sin kx]$$

The application of this solution to the PVC model will be shown in Section 3.2.2.

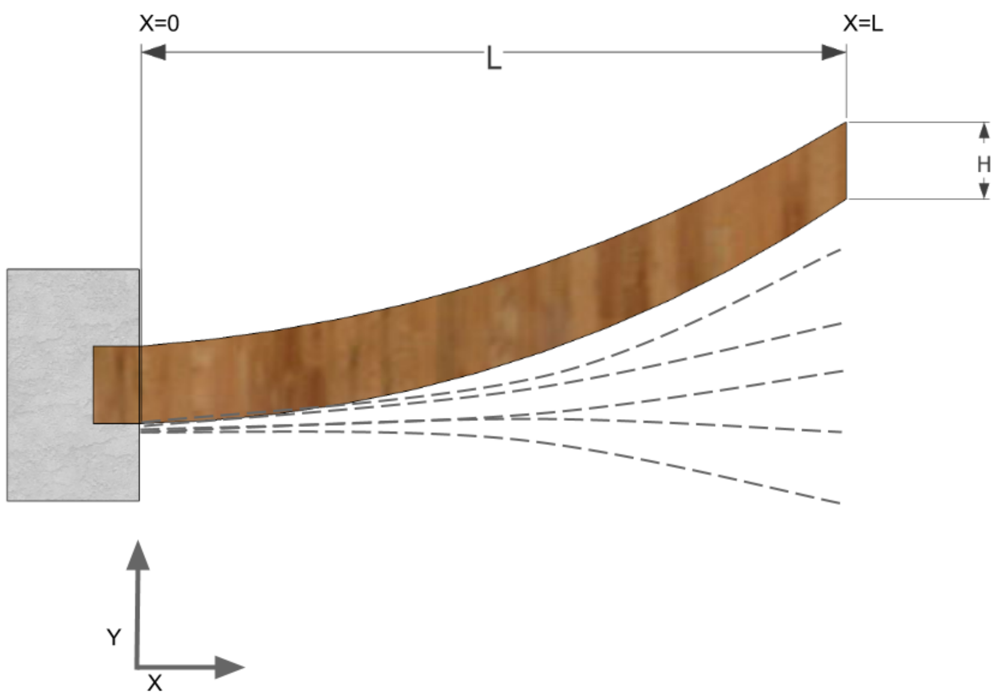

Figure 3.1: Bar held at one end.

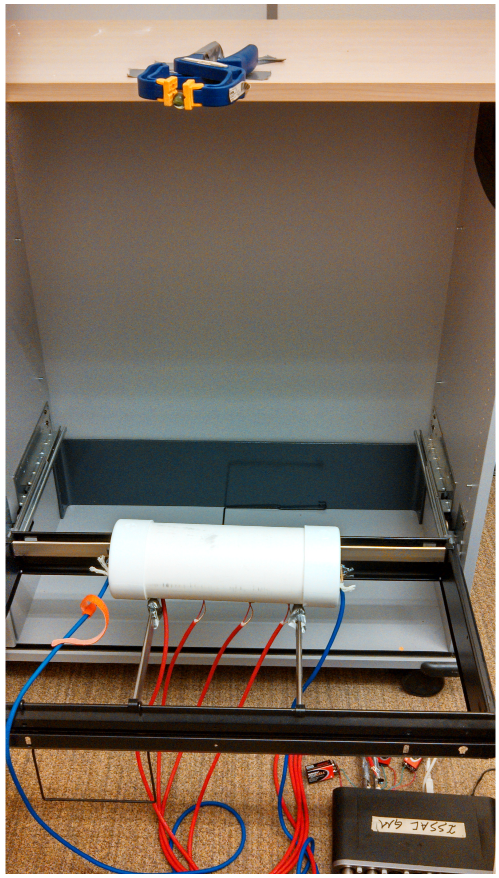

Figure 3.2: Setup for the acoustical measurements on the PVC model.

3.2 Simplified model using PVC pipe

3.2.1 Setup

Here we try to obtain a measurement of the impulse response of a PVC prototype model of the teponaztli to characterize the acoustic behavior of the instrument. The assumption being that the model behaves like a linear time-invariant (LTI) process whose transfer function can be characterized by the impulse response. A system was constructed to drive the model with a relatively consistent impulse-like signal and record the response. Transfroming the response yields an approximation of the frequency response. The experiments were performed on a PVC prototype model of the instrument for simplicity and ease of construction. This way the response could be measured while incrementally modifying the PVC cylinder to resemble the teponazlti, starting with it completely closed, then one tongue cut out, then both and finally an aperture below the tongues. The last step was to try and model the resonance cavity of the teponaztli as it would appear in a traditionally constructed teponaztli in hopes of getting a stronger acoustic sound.

The experiment setup (See Figure 3.2) involved dropping a marble onto a PVC pipe which is closed at both ends. The marble was clamped with a quick release vice and dropped from the same height each time to make every impact as identical and repeatable as possible. To take the measurements of the model I built two electret microphones with phantom powered balanced preamps. The first prototype proved contact microphones to be insufficient so I wanted the prototype model to explore the use of electret microphones. These measurements were setup to determine the best placement of the microphones for the next version of the instrument. Holes of the diameter of the microphone capsule were cut into the tube so that they would be pressed flush with the walls (See Figure 3.3). In this manner I was able to get the response measurements at various locations of the bottom and sides of the tube.

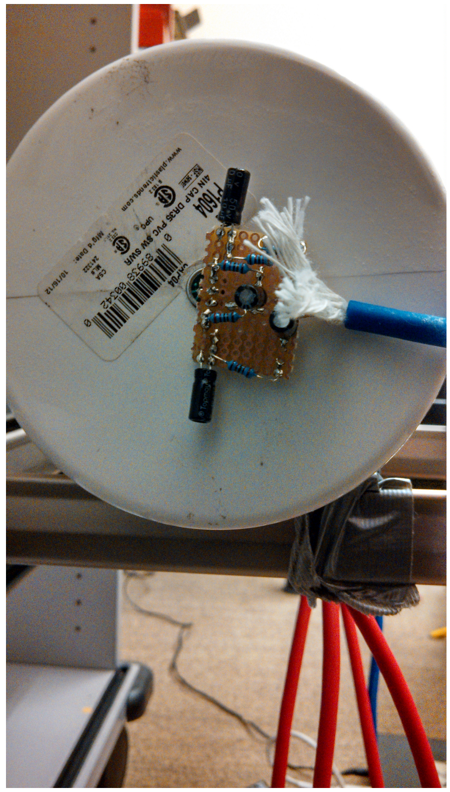

Figure 3.3: The microphone is placed flush with the endcap.

3.2.2 Measurements

Using the dimensions from one of the tongues, $L = 10cm, H = 0.317cm$, the first three frequencies of vibration for the PVC tongue were calculated using equation 3.1 to be $f_1 = 24Hz, f_2 = 152Hz, f_3 = 424Hz$. The one-dimensional model of the tongue from Section 2.1 results in an $f_1 = 850Hz$. The measurement from the embedded microphones in response to the marble drop shows the resonance peaks to start at $141Hz$ in Figure 3.4. The discrepancies between the calculations and the measurement from the marble drop could be attributed to the PVC tongue being far from an ideal bar because of its curvature and the fact

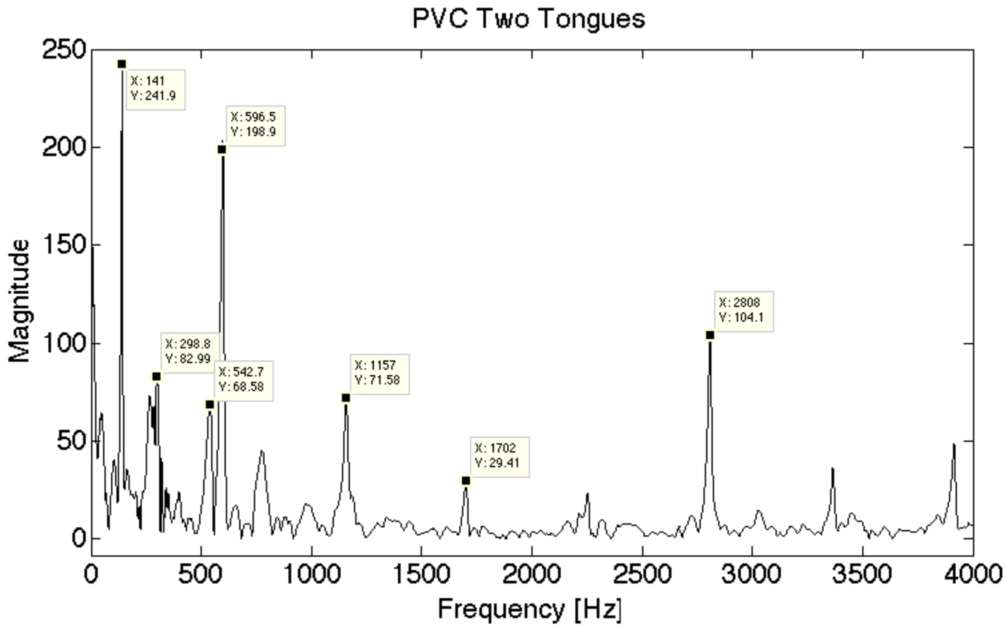

Figure 3.4: Measurement of PVC tube with two tongues.

that the measurement is a combination of the resonance of the tube and vibration of the tongue. To take into account the resonance from the PVC tube, the PVC model was approximated to be a one-dimensional cylinder (See Figure 3.5), where the dimensions of the cylinder are $l = 30cm, h = 10cm$ so the first fundamental would be:

$$f = \frac{c}{\lambda} = \frac{340m/s}{0.6m} = 567Hz$$

We can see in Figure 3.6 that the first peak of the marble drop on a closed PVC pipe is very near the theoretical. The $27Hz$ difference could be attributed to the assumption that the speed of sound is exactly $340m/s$ in the calculation of equation 2.1.

When taking these measurements I was also curious to see if there was

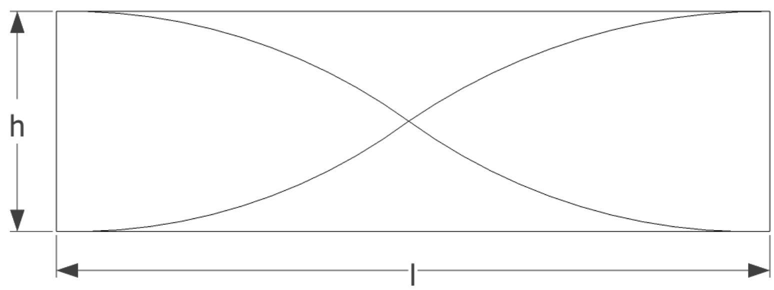

Figure 3.5: Cross section of the PVC tube showing longitudinal pressure nodes and anti-node corresponding to the fundamental frequency of a one-dimensional cylinder.

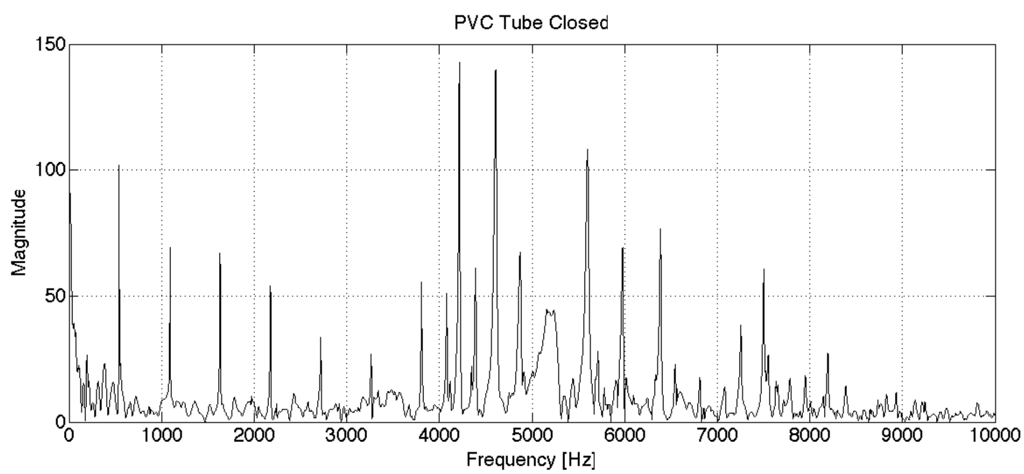

Figure 3.6: Measurement of a marble drop on the closed PVC tube. The first peak is at $540Hz$ which relates well to theory, but the high energy above 4kHz was unexpected.

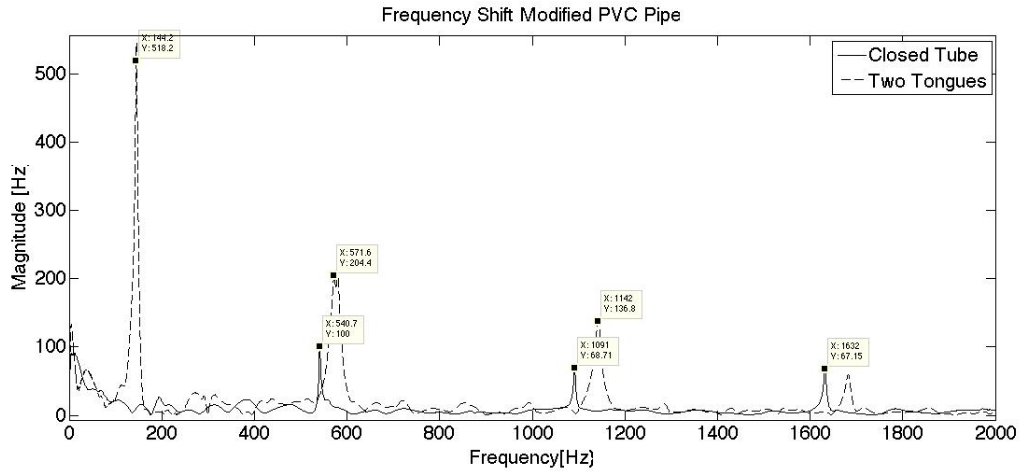

Figure 3.7: Closed PVC tube versus tube with two tongues.

evidence of transverse modes of a cylinder, waves propagating along the diameter of the tube, but knew this would be difficult especially because they are highly attenuated [FR98]. If they are calculated for the PVC tube in the same way as was done for the longitudinal waves then the result is $1673Hz$ for the first mode. The dimensions of the PVC tube have a 3:1 length-to-diameter ratio which makes it difficult to see if the transverse modes are contributing to the peak at around $1673Hz$ because this could be the first transverse frequency or the third longitudinal partial.

3.2.3 Examining the Measurements

It is interesting to note the shift in the frequencies when the closed tube is modified with the tongues (See Figure 3.7). This could possibly be characterized by modeling the tongue slits as if they were sound holes on a wind instrument, but this is beyond the scope of this thesis. Taking this shift into account, the measurements do seem to follow the one-dimensional cylinder calculations. And the peak that appears at around $140Hz$ is most likely due the vibration of the tongue since it is absent from the closed PVC measurement. What remains unexplained is the high energy above 3kHz which is clearly shown in Figure 3.6. This may be a result of the physical shock on the microphones from the marble impact. In retrospect, the measurements should have been repeated after adding shock absorption to the microphones in order verify the hypothesis. For the next version of the instrument, as will be shown in the next section, the microphones were padded with anti-static foam to absorb shock.

Out of the six placements of the electret microphones on the PVC pipe (left and right ends on axis with the center of the pipe, and four places on the side of the pipe) it was found that location with peaks of greater magnitude were on the ends of the pipe. These were expected because they are anti-nodes and this became the location chosen for embedding the microphones in the second prototype.

In addition to informing the design and construction of the next version of the electroacoustic teponaztli, the PVC prototype model also served as a first attempt at the goal of differentiating between percussive strikes. The approach was to separate the excitation from the resonance of the model. This assumes that the excitation remains constant and what changes is the resonance/filter of the instrument. The marble drop being the excitation was hoped to be close to an ideal impulse such that the measurements would approximates an impulse response of the drum at the strike point. In this manner the transfer function could be used to extract a percussive hit with a mallet or other stick at the strike point by inverse filtering. Unfortunately, the marble drop was not a close enough approximation of an impulse or the model was not a close enough approximation to an LTI system so I was unable to extract the source of excitation. Therefore, other signal analysis methods for differentiation were pursued and will be shown in Chapter 4 including how an external sensor can be used to track the strike point.

3.3 Second Prototype: traditionally constructed teponaztli

Figure 3.8: Second prototype.

A second prototype of the electroacoustic teponaztli was crafted with the goal of improving the acoustic sound and implementing the results of the acoustic measurements made on the PVC simplified model (see Chapter 3). The wood was from the same eucalyptus tree as the first prototype, but 7" in diameter and 20" in length. A problem with repurposing these pieces and a downside to using eucalyptus wood in general is that it cracks very easily. Because they were not left to dry in a cool place with small temperature variations there are noticeable cracks running lengthwise, some of which had to be repaired using sawdust and glue. Also, the location of the tongues was chosen based on the part of the wood that had minimal fracturing. I have been able to obtain tree trunks from the La Jolla area recently which I plan on using to make more teponaztlis in the future. This time I am letting them dry in a storage closet and with their bark intact to see if this minimizes the cracking. Both prototypes were treated with tung oil once the carving was complete which helped to close the cracks as the wood absorbed the oil. This finish was chosen instead of a synthetic laquer to maintain a natural look.

A rectangular cavity, 3.5" wide and 13" long was carved underneath the position where the tongues would be made. This was done by hand with a wood chisel and mallet. On the tongue side the trunk was planed. This is a derivation from the museum model whose tongues retain the curvature of the tree trunk, but I thought that it would facilitate comparing the results with the theoretical fundamental resonance of a bar held at one end.

The tuning in this case was achieved by finely adjusting the thickness of the tongues, holding their lengths equal. Each tongue is about 6" long and 3-

Figure 3.9: Underside of the teponaztli.

1/4" wide. When one of the two tongues began to sound particularly well (using a medium hard knit percussion mallet), its closest note was measured to be B4 using a guitar tuner. It was found that reducing the thickness of the other tongue increased its frequency which is counter to equation 3.1. This was done to go from F4# to G4# and end up with a minor third interval.

From Chapter 3 it was shown that the best placement for embedding the electret capsules is on the inside of the resonant cavity and on axis. The same balanced phantom powered microphones used in the PVC experiments were the ones mounted on the instrument. The circuit was wrapped in anti-static foam and electrical tape and glued to the inside of the resonant cavity at each end and on axis (See Figure 3.10). In a future version I would create a recessed mounting for the microphone such that it is flush with the wall and thus reduces any interference it may cause to the resonance. Two male XLR panel connectors were recess mounted on the side of the instrument. Once the instrument was built, the next step was to characterize its acoustics and use that information to turn it into a computer-

Figure 3.10: Microphones mounted on the teponaztli.

extended instrument.

Chapter 4

Extracting Control Values from the Percussionist

From working on *Dos Mil Doce* it was made evident that having to look at the computer screen and move away from the drum to change the knobs of a MIDI controller detracts from the performance of the teponaztli. Being able determine where the performer has struck the instrument affords a parameter with which to control the processing without interfering with the player's performance. It is also something that should have a one-one relationship such that the audience can perceive that a process is taking place and not just"magic" [SJ93]. In the shared instrument example from section 2.3 I discussed how the control could be given to another performer through a TCP/IP network, but the idea for the current version of the teponazlti is to move the control back to the percussionist by extracting information from their performance using signal analysis and an infrared sensor.

4.1 Capturing Gestures Using an Infrared Sensor

Although the case can be made in favor of using the acoustic sound of an instrument instead of physical capture as a source of control for electronic processing [Puc], I explored using an infrared sensor to capture the gestures of a percussionist and turn them into control signals for audio/video effects. The infrared sensor that I am using to track the mallets is called LeapMotion and I chose it because it can capture the three-dimensional position of hands, fingers and sticks. This is similar to the system that Jaime Olivero built for his virtual drum, the Silent Drum[1], in which the performer's hands interact with a mesh fabric, are recorded by a video camera and are converted into virtual movements [LRE11]. His software calculates the distance between each finger and hand present on the screen and uses this information in his synthesizer. Even though the Silent Drum produces entirely synthesized sounds, and the teponaztli processes its own acoustic sound, there are some concepts that can be applied such as the Z axis being used for dynamics. Also, the distance between fingers is another parameter that is used in the Silent Drum that could be used with the teponazlti if the performer chooses to play with their hands instead of mallets.

I believe that infrared sensors, like the LeapMotion, can make excellent trackers because they can be used with any percussion setup where there is space

[1] http://www.jaimeoliver.pe/instrumentos/silent-drum

for the sensor between the object being struck and the percussionist. I received a LeapMotion device as an early developer before it was brought to market and began working with a python script from the Leap API library that sent information from each captured frame using the Open Sound Control (OSC) protocol. These messages would then be received and routed in Pd. The different types of messages are finger position (x, y and z), finger displacement (dx, dy, dz), palm position and palm displacement. Here is a snapshot of the output from the python script:

```
Drop lost RealFingerTracker:5
Zeroing lost RealFingerTracker:2
Zeroing lost RealFingerTracker:2
Zeroing lost RealFingerTracker:2
Saw 101 frames; Sent 6246 messages in 0:00:01.004537
```

It shows that there are about 100 frames per second (fps) and that there were dropped fingers and zeroing losses which, from experience, happened when all ten fingers were being tracked. With only one hand the messages per second are around 1000, but over six times as much with both hands and all fingers, 6246. I wondered if all these messages are being received in Pd through OSC in a timely manner and if I could improve upon this? Therefore, I wrote a Pd external object which receives the frame messages directly from the LeapMotion API without having to be relayed through OSC. I was able to output the time lapsed between each frame for the leap Pd external and found that each frame comes at about 8600μsec (about 116fps). So the difference in frame rate is negligible, but what about the time between each message? When putting all 10 fingers in the detection area, the finger position update time using OSC messages averaged to 15ms (routing 10 parameters), while the Pd external has an average of 5ms between new floating point position. So with the Pd external the data is received as fast as a the frame rate, whereas there is a delay when using OSC. This was expected because parsing the 1000+ OSC messages was bound to cause messages to be delayed.

The LeapMotion API has the capability of detecting what it calls a tool, which refers to a pencil shaped object at the end of a closed hand. This would be perfect for tracking the mallets, but unfortunately I have not been successful in accessing these messages. I believe it is because I am only receiving the frame data when a hand is detected, but since the hand is closed in this case I would need to setup a different call-back routine for the tool detection. The C-wrapper classes that I am using (because the LeapMotion API is written in C++ and the Pd externals are in C) does not have such a routine yet. In the meantime, if the device is placed between the body and the hands it sometimes catches the end of the mallet and thinks of it as a finger in which case I am able to receive the position in this manner. Ultimately, I would need the tool detection data in order for the sensor to be used in a performance setting.

4.2 Differentiating Percussive Strikes

In addition to using an infra red sensor such as the LeapMotion to capture mallet position I investigated using the information from the microphone signals to determine where the object was struck. First I took an impulse response from a mallet strike on each tongue (See Figure 4.1). The frequency domain plots show a

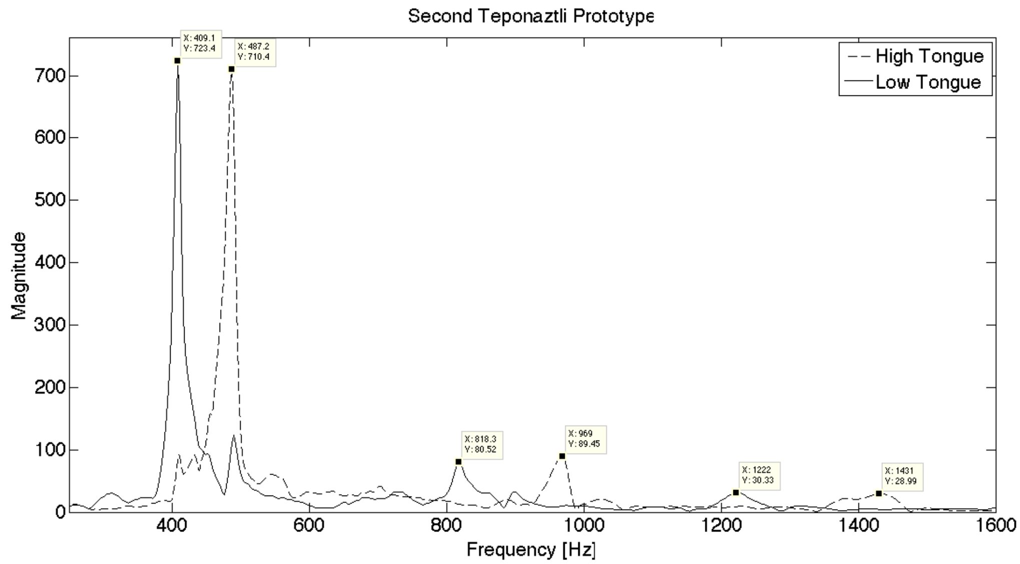

Figure 4.1: Frequency Spectrum of the two notes on the second prototype.

strong resonance at the fundamental frequency for each tongue and two other partials. Low tongue: $409Hz, 818Hz, 1222Hz$; High tongue: $487Hz, 969Hz, 1431Hz$. Beyond the $1431Hz$ peak there was not enough signal to noise ration to make out any other partials. Using this information I constructed a small network of bandpass filters with center frequencies at each partial and high Q. The idea was to use envelope followers from the output of the filters and take a snapshot of each strike and based on the ratios between the partials determine where the drum was struck. To simplify the analysis these were the different strikes to be captured:

1. left tongue mallet

2. right tongue mallet

3. side of the teponaztli mallet

4. left tongue stick (other end of the mallet)

5. right tongue stick

6. side of the teponaztli stick

The problem with using only the six bandpass filters was that extraneous noise, a hand clap for example, lit up the filters. Templates of the relative amplitudes of each partial would have to be created. To be assured of the features that could be used to differentiate between strikes the spectral stability for each strike was plotted (See Apendix B.1). Each of the six strikes was recorded 50 times and then the normalized average of the spectrum was plotted as well as one standard deviation above and below the mean. There was very little deviation from the peaks in this case, but then it was repeated varying dynamics and position on the tongues (See Apendix B.2). It was clear that there was enough distinction between the features and relative peak heights to distinguish each of the six strikes.

Instead of creating my own template with which to test for these features, I used the native Pd object Bonk˜ which takes care of onset detection and spectral envelope measurement. I began by setting it up with these initial conditions:

bonk˜ -npts 1024 -hop 256 -halftones 1 -nfilters 22 -overlap 2 -firstbin 9.5

for 1024 sample analysis and a hop size of 256 samples with 22 filters of one halftone bandwidth (and an overlap of 2). The first analysis bin has a center frequency of $9.5 \times 43.07Hz = 409Hz$, and the last bin has a center frequency of

$33.41 \times 43.07Hz = 1439Hz$. Bonk also has a learning feature where it develops a template from a specified number of strikes and then when learning is turned off, each subsequent attack's euclidean distance to each template is calculated and the "cooked" value is output which is the template with the closest distance. With this feature I was successful in differentiating between the first three objectives (mallet left, right and side) and minimizing the effects of extraneous noises.

What Bonk˜ was not able to do initially was distinguish between mallet and stick strikes. The plots in Appendix B are made by taking the FFT of the entire strike, but it was found that the difference between stick and mallet could be best obtained in the attack. By looking at the first 512 samples of the stick versus mallet (See Figure 4.2) it is clear that the stick has energy in the region around 4kHz and the mallet doesn't. So the relationship between the first three peaks and this region combined could be used to differentiate between the six types of strikes on the teponaztli. Therefore, I tried extending the number of bins for Bonk˜ to reach the 4kHz area, but this made it more difficult to distinguish between any of the strikes. The solution was to split up Bonk˜ into two objects, one with the same number of bins as before and another with bins starting at 3446Hz up to 5795Hz. Now I was able to use the learn function with these objects to obtain a clear "cooked" value for each of the six strikes. Further work could be done to detect the location of the strikes on the tongue more precisely (closer to the free end or closer to the joined end, for example) by characterizing the acoustics of each tongue in detail.

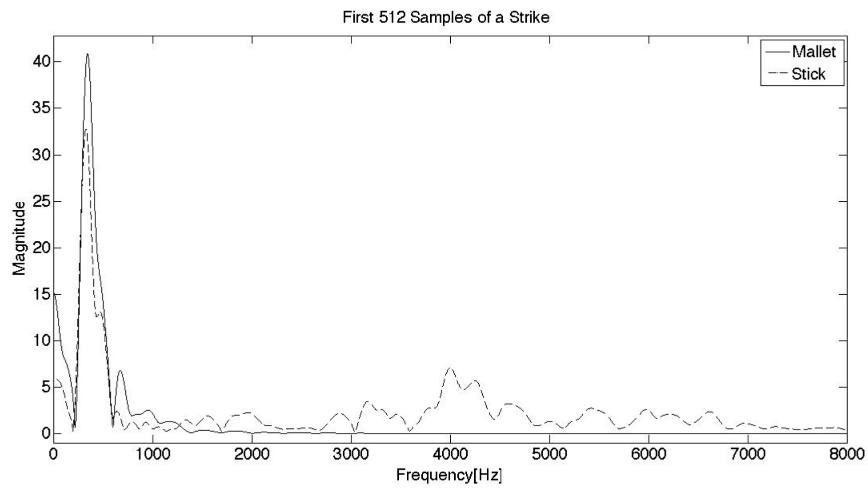

Figure 4.2: Low Tongue Mallet VS Stick Attack.

4.3 Robustness

The detection Pd patch from the previous section was created in a quite setting. To prepare the instrument for a performance setting a few robustness tests were run. As a starting reference, a typical mallet strike measured >100dB SPL[2] and a stick strike >90dB at the entrance to the resonant cavity of the instrument. The patch was able to handle 73dB SPL of white noise before it began to give false positives. This number rose to 93dB when the source is a low passed white noise with a cutoff frequency of 100Hz, which was tested to account the high energy of sub woofers in a concert scenario. And only about 60dB with an impulse like a hand clap which was the same problem I was having before with the series of bandpass filters. At that level any another percussive source or instrument with

[2]The SPL A-weighted was measured with an Android Motorola Razr M using the Sound Meter App.

similar resonance frequency would trigger false positives. Fortunately I was able to bring down the input gain of the microphones to a point where the hits could still be detected while rejecting extraneous noises. White noise now had to be about 100dB and a hand clap >95dB in order to trigger a false positive.

Chapter 5

Conclusion

5.1 Embedded Systems Integration

A last step in the transformation of the teponaztli into an electroacoustic instrument is to have it become an entirely self contained instrument, meaning that all the audio signal processing takes place on an embedded system that is incorporated on the instrument. Similar to the SABRe clarinet and the Zeta violin pickup. I would propose moving the granular delay, strike estimation and convolution processing mentioned in earlier sections to a DSP chip. As a first step in this endeavor I have begun work with the Analog Devices Blackfin BF537 DSP. I must reveal that the choice for this chip was not particularly for any of its specifications, but because I was already in possession of the evaluation board. The Rasberry Pi[1], a miniature computer, may actually be more suitable because

[1] http://www.raspberrypi.org/

it runs linux so that the Pd patches would not have to be converted to DSP code and the LeapMotion device could be seamlessly integrated as well[2]. But for the time being and as an exercise I chose to proceed with the Blackfin BF537 DSP which has a 600MHz clock, 32-bit wordlength and 132K on-chip memory.

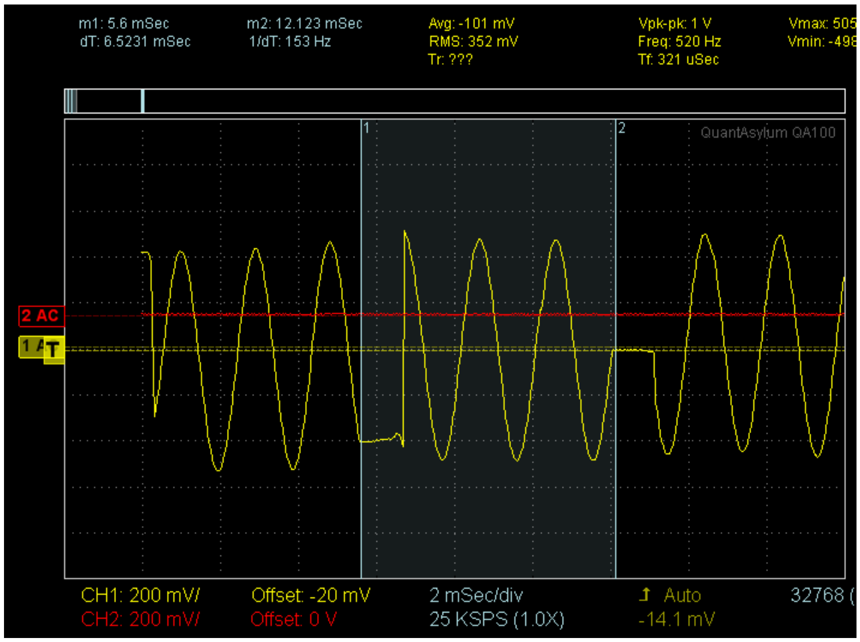

Figure 5.1: Talkthrough output of a sine wave input to the Blackfin DSP which shows how input samples were lost every frame.

The ADC and DAC are 24-bit off-chip peripherals whose data had to be converted to fractional 16-bit numbers for the FFT functions. Direct memory access (DMA is used to transfer the samples into memory for processing. The interrupt requests are set at the sample rate of 48kHz. In this per sample DMA method I was buffering the input and performing a four-time overlap add with an FFT size of 1024 samples and a frame size of 256 samples (the code can be found in Appendix C). Unfortunately all the computations that were done every frame

[2]The LeapMotion API is available for linux.

took longer than the time between samples so input samples were not received (See Figure 5.1). A work in progress is to set the DMA for a blocksize of the ADC and DAC data that is large enough (instead of per sample) that would allow the computations to be run between each DMA data transfer.

5.2 Teponaztli Transformed

After a first prototype of the instrument was constructed, signal analysis measurements on a simplified PVC prototype model and feedback from performing *Dos Mil Doce* were methods vital to the improvement of the instrument and the construction of a second version. Attempting to extend control of the instrument telematically was an experiment which not only resulted in interesting observations, but also led to the decision to place the control of effects parameters in the hands of the percussionist. But this control had to be extracted from the natural movements of the player so as not to detract from their performance of the instrument. Using the infrared sensor, LeapMotion, to track the drumsticks without interfering with the percussionist's movements remains a work in progress. Feature reliability measurements performed on the second version showed that the goal of differentiating among six different types of drum hits could be accomplished in real time by comparing the ratios of resonance peaks. This was achieved in Pd with the help of the Bonk~ object and by splitting up the task of detection into two frequency regions. I believe that the transformation of the teponaztli into an electroacoustic computer-extended instrument has been a successful merging of a traditional instrument with current computer music technologies.

Appendix A

Algorithmic Composition Code

A.1 AthenaCL

```
Testing Genetic Algorithm:
timute b c d
tio a
tie x6 sampleSelect,nahuatl.aiff,oc
tie r gr,((1,1,1),(1,1,1),(1,1,1)),1,1,0
eln
ticp a genetic
tirm a

gaRhythm, ((1,1,+),(1,1,+),(1,1,+)), 1, 1, 0, orderedCyclic,20
                    ((1,1,+),(1,1,+),(2,2,+))
                    ((44,45,+),(4,4,+),(5,5,+))
                    ((6,5,+),(1,1,+),(5,5,+))
                    ((18,18,+),(4,3,+),(1,1,+))
                    ((1,1,+),(18,12,+),(2,2,+))
                    ((5,5,+),(8,8,+),(8,10,+))
                    ((8,10,+),(2,2,+),(10,10,+))
                    ((5,5,+),(1,1,+),(3,4,+))
                    ((1,1,+),(18,18,+),(2,1,+))
                    ((2,1,+),(1,1,+),(3,3,+))
                    ((2,1,+),(3,3,+),(2,3,+))
                    ((2,3,+),(4,2,+),(1,1,+))
```

```
((9,9,+),(2,3,+),(2,1,+))
((4,6,+),(4,4,+),(5,5,+))
((2,2,+),(1,1,+),(2,2,o))
((2,4,+),(2,2,+),(5,1,+))
((2,1,o),(2,2,+),(2,2,+))
((4,4,+),(2,1,o),(1,1,+))
```

The gaRhythm by itself was not creating the gradual progression I sought out so then I tried changing the sample start parameter from random to a Fibonacci series.

```
Testing Fibonacci Object:
timute genetic
tio b
tie x6 sampleSelect, nahuatl.aiff,oc
tie x5 fibonacciSeries,1,5,0,30,oc

Final Commands:
tie x5 cyclicGen,up,0,20,0.5
tie t 0, 60
ticp b intro
tirm b
tio c
timute c
tie x6 sampleSelect, spanish.aiff,oc
tie x5 cyclicGen,up,0,20,0.5
tie t 20, 60
ticp c spanish
tirm c
eln

tio d
tie r iterateRhythmWindow,
((loop,((4,3,+),(4,3,+),(4,2,o),(8,1,+),(4,2,+),(4,2,+)),
orderedCyclic), (convertSecond,
(randomUniform, (constant,1.5), (constant, 4)))),
(basketGen, randomChoice,(-3,6,-1,15)), orderedCyclic
tie x6 sampleSelect,(nahuatl.aiff,spanish.aiff),oc
timute intro spanish d
ticp d iterative
tirm d
```

```
eln

tin reverb 30
tie x5 sampleSelect,(nahuatl.aiff,spanish.aiff),oc
tie r l,(1,1,1)
tie x0 cyclicGen,up,0,20,0.5
eln
(Renamed to run8_reverb.aif)
tie t 60,120
timute intro spanish iterative genetic
tio iterative
tie t 50,120
tio genetic
tie t 60,120
eln
```

A.2 Sphynx Speech Recognition

```
#include <pocketsphinx.h>
#define MODELDIR "/pocketsphinx-0.7/model"

int
main(int argc, char *argv[])
{
ps_decoder_t *ps;
cmd_ln_t *config;
FILE *fh;
char const *hyp, *uttid;
        int16 buf[512];
int rv;
int32 score;

config = cmd_ln_init(NULL, ps_args(), TRUE,
     "-hmm", MODELDIR "/hmm/en_US/hub4wsj_sc_8k",
     "-lm", MODELDIR "/lm/en/turtle.DMP",
     "-dict", MODELDIR "/lm/en/turtle.dic",
     NULL);
if (config == NULL)
return 1;
ps = ps_init(config);
```

```
if (ps == NULL)
return 1;
//open the audio file
fh = fopen("goforward.raw", "rb");
if (fh == NULL) {
perror("Failed to open goforward.raw");
return 1;
}
//make sure it is the start of the file
        fseek(fh, 0, SEEK_SET);
        rv = ps_start_utt(ps, "goforward");
if (rv < 0)
return 1;

//timer variables declared and initialized
float samples;
float seconds;
samples = .0;
seconds = .0;
        while (!feof(fh)) {
         size_t nsamp;
         nsamp = fread(buf, 2, 512, fh);
             rv = ps_process_raw(ps, buf, nsamp, FALSE, FALSE);
hyp = ps_get_hyp(ps, &score, &uttid);

//ps_get_hyp returns a char if found a word or else NULL
if(hyp!=NULL)
{
printf("hyp is: %s\n",hyp);
//change from samples to seconds, 16000 is sampling rate
seconds = samples/16000.0;
printf("Time processed at: %f\n", seconds);
}
samples += 512.0;
        }

rv = ps_end_utt(ps);

if (rv < 0)
return 1;
hyp = ps_get_hyp(ps, &score, &uttid);
if (hyp == NULL)
return 1;
```

```
//output final phrase
printf("Recognized: %s\n", hyp);

fclose(fh);
        ps_free(ps);
return 0;
}
```

And an excerpt of the output:

```
hyp is: go
Time processed at: 0.608000
hyp is: go forward
Time processed at: 1.088000
hyp is: go forward ten
Time processed at: 1.440000
hyp is: go forward ten meters
Time processed at: 1.952000
```

And now using this information in AthenaCL (1.952 seconds was the end ofthe words so it is not used and the start of the phrase is at 0.5 seconds):

```
emo cn
tin forward 32
tie x6 sampleSelect, goforward.wav, oc
tie r l,(1,1,1)
tie x5 basketSelect,(0.5,0.608,1.088,1.44)
eln
```

Appendix B

Spectral Feature Reliability

The following are plots of the six different strike categories with a sampling size of 50 hits. The data is normalized and then the avergage and average $\pm$one standard deviation (σ) are plotted.

B.1 Homogeneous Strikes

The strikes for these plots were performed as identical to one another as possible.

B.2 Varied Strikes

The following strikes were varies with respect to force (dynamics) and location within the specified strike area.

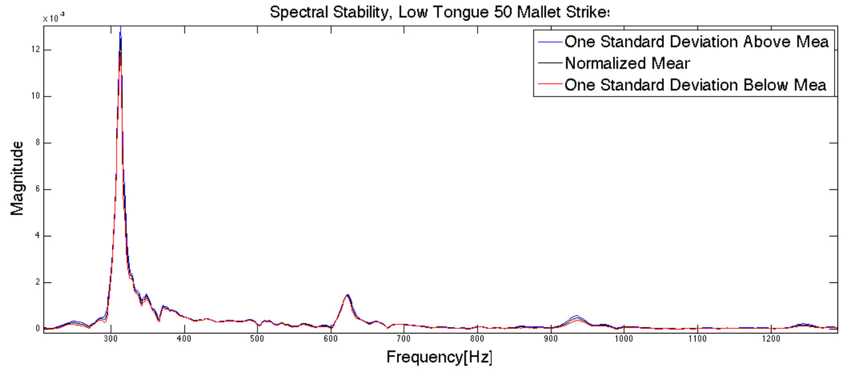

Figure B.1: Low Mallet

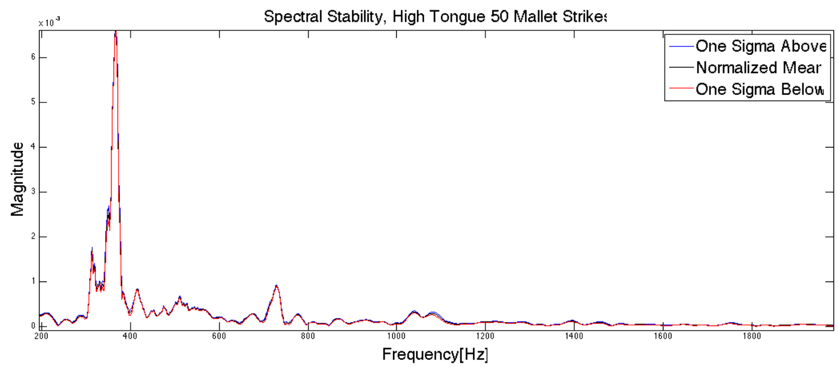

Figure B.2: High Mallet

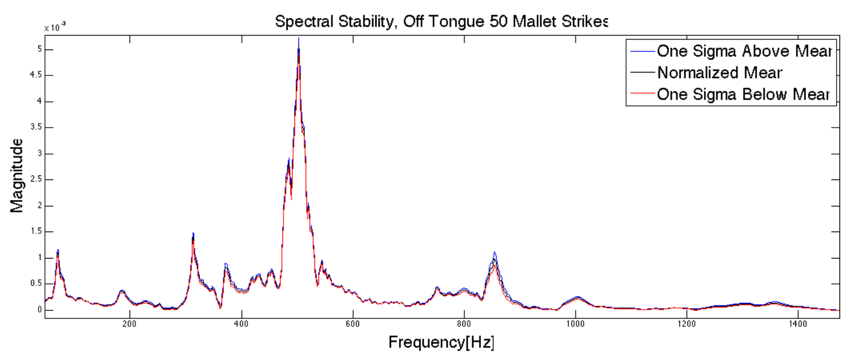

Figure B.3: Body Mallet

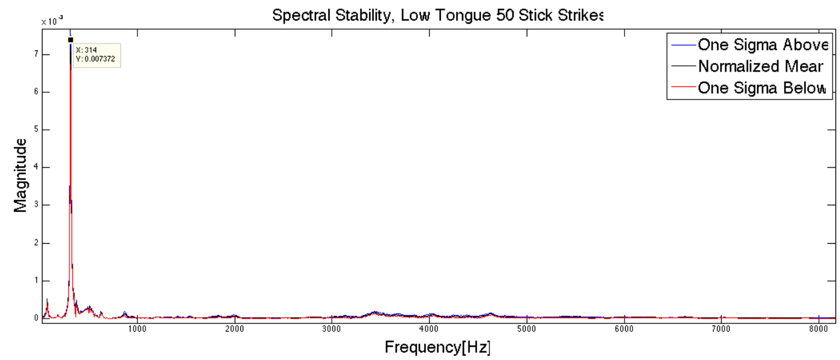

Figure B.4: Low Stick

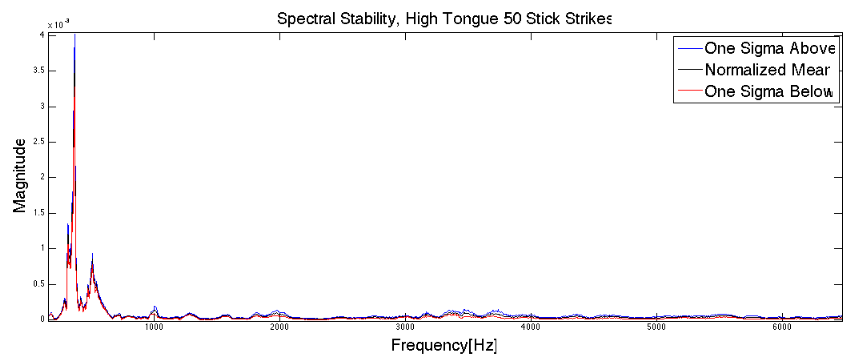

Figure B.5: High Stick

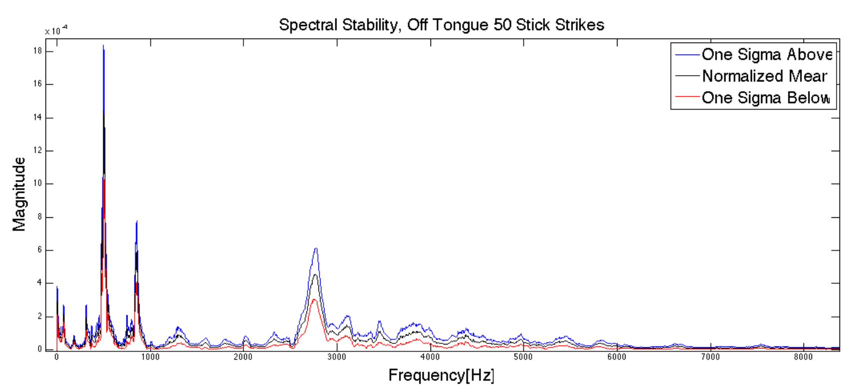

Figure B.6: Body Stick

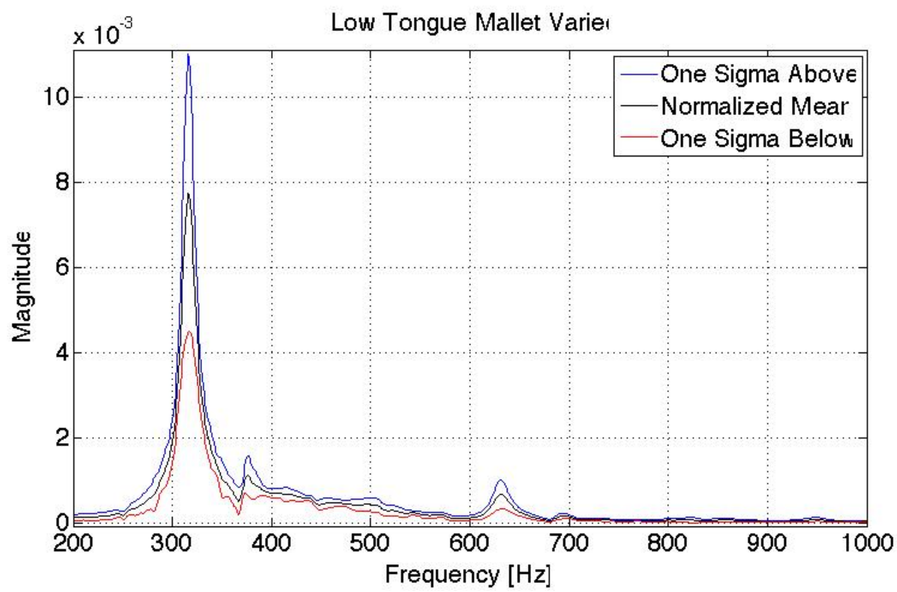

Figure B.7: Low Mallet Varied

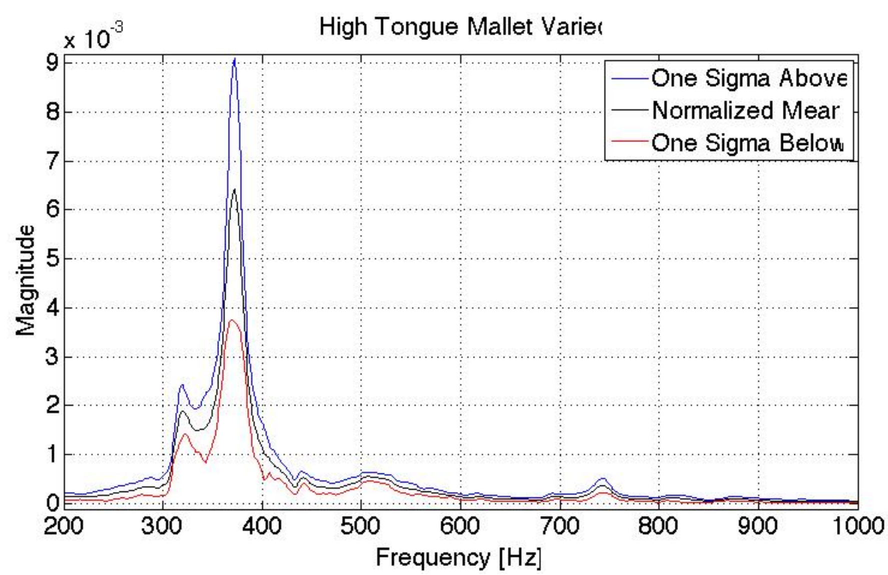

Figure B.8: High Mallet Varied

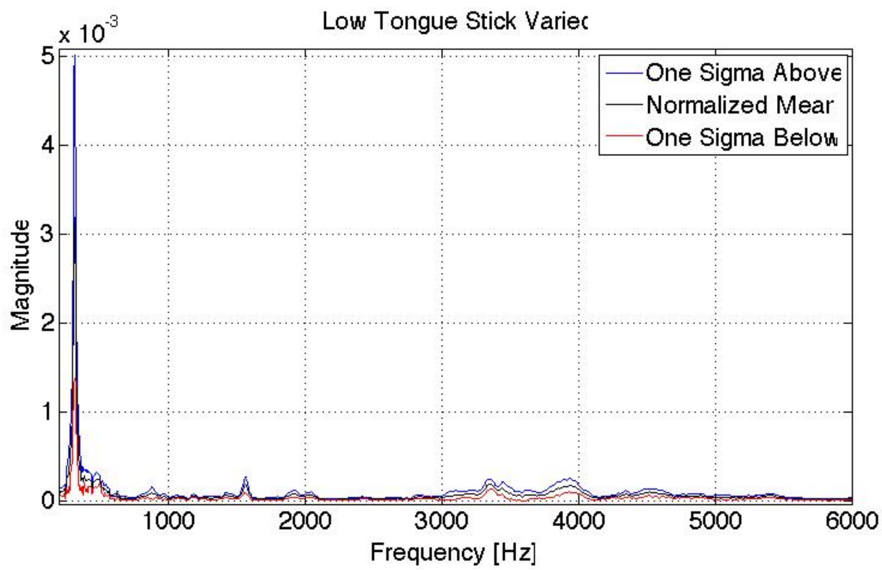

Figure B.9: Body Mallet Varied

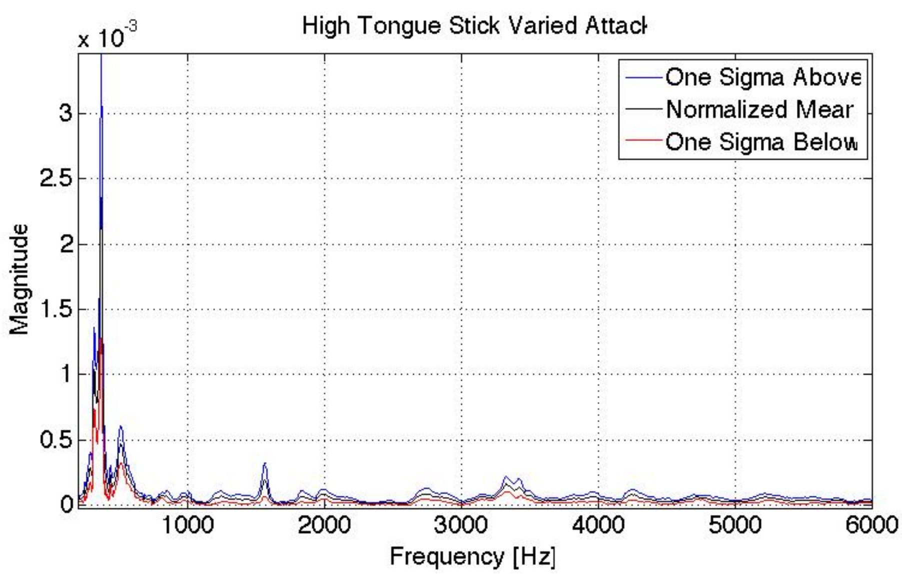

Figure B.10: Low Stick Varied

Appendix C

Convolution Code

The following is the DSP Code for implementing convolution on the Blackfin BF537 DSP.

```
void Process_Data(void)
{
    if (n>BLOCKSIZE-1){
        //reset block counter
        n=0;

        //shift the input fft buffer a blocksize
        memcpy(infftbuff,infftbuff+BLOCKSIZE, N_FFT*2);
        //load in the new samples
        memcpy(infftbuff+(3*BLOCKSIZE),inbuff,BLOCKSIZE*2);

        //window the input fft buffer
        for (i=0;i<N_FFT;i++){
            winfftbuff[i]=mult_fr1x16(infftbuff[i],window[i]);
            }

        //Compute FFT
        rfft_fr16(infftbuff, temp, outfftbuff, twiddle, 1 , N_FFT, 1, 1);

        //TO DO: COMPLEX MULTIPLY HERE

        //inverse FFT
```

```
    ifft_fr16(outfftbuff, temp, ifftout, twiddle, 1, N_FFT, 1, 1);

    // load the output buffer that feeds the DAC
    memcpy(outbuff, convout, BLOCKSIZE*2);
    //shift over before the ovelapp add
    memcpy(convout, convout+BLOCKSIZE,N_FFT*2);
    //clear the last BLOCK in the convout buffer
    memset(convout+BLOCKSIZE*3,0,BLOCKSIZE*2);

    for (i=0;i<N_FFT;i++){
        //window ifft output and overlapp add
        wifftout[i]=mult_fr1x16(cabs_fr16(ifftout[i]),window[i]);

        convoutintermediate=add_fr1x16(convout[i],wifftout[i]);
        convout[i]=convoutintermediate;
    }
}

// Load in a new sample in to the input buffer
inbuff[n]=(short)(iChannel0LeftIn >> 8);
// send out a processed sample to the DAC
iChannel0RightOut=(int)(outbuff[n] << 8);
//increase the block counter

n++;
}
```

Bibliography

[BF03] Tina Blaine and Sidney Fels. Contexts of collaborative musical experiences. In *Proceedings of the 2003 conference on New interfaces for musical expression*, pages 129–134. National University of Singapore, 2003.

[BP00] Tina Blaine and Tim Perkis. The jam-o-drum interactive music system: a study in interaction design. In *Proceedings of the 3rd conference on Designing interactive systems: processes, practices, methods, and techniques*, pages 165–173. ACM, 2000.

[FKTM04] Sidney Fels, Linda Kaastra, Sachiyo Takahashi, and Graeme McCaig. Evolving tooka: from experiment to instrument. In *Proceedings of the 2004 conference on New interfaces for musical expression*, pages 1–6. National University of Singapore, 2004.

[FR98] Neville Horner Fletcher and Thomas D Rossing. *The physics of musical instruments*. Springer Verlag, 1998.

[LRE11] Oliver La Rosa and Jaime Eduardo. A computer music instrumentarium. 2011.

[Moo90] F Richard Moore. *Elements of computer music*. Prentice-Hall, Inc., 1990.

[Par05] Gabriel Pareyón. El teponaztli en la tradición musical azteca. 2005.

[Puc] Miller Puckette. Grafting synthesis patches onto live musical instruments.

[Puc07] Miller Puckette. *The theory and technique of electronic music*. World Scientific, 2007.

[SJ93] W Andrew Schloss and David A Jaffe. Intelligent musical instruments: The future of musical performance or the demise of the performer? *Journal of New Music Research*, 22(3):183–193, 1993.

[SS] Sébastien Schiesser and Jan C Schacher. Sabre: The augmented bass clarinet.