Andrew Jarowenko
Electronic Musical Instrument Design
Professor Paul Lehrman
May 7, 2015

Figure 1

The Sonic Pyramid

(AKA The All-Seeing Eye of Sound)

The sonic pyramid is a collaborative music performance interface meant to be shared by three players[1]. Each player has a side of the pyramid with which to make music and jam with the other players. The idea for the project came from Mike and Tom and was started as their project-1. Greg and I joined forces with them and continued work on the pyramid for project-2.

[1] Though it can successfully and enjoyably be played by fewer or more players

Design of the Instrument

Initially, each of the three faces of the pyramid was to contain five push-buttons and two circular soft-potentiometers. The push-buttons would trigger samples or synthesizer notes, and the soft-pots would modulate relevant parameters. This was a nice initial concept, but not very mature or interesting. Five notes don't allow very much room for creativity in performance.

When Greg and I joined the team, the stated goal was to increase the variety and complexity of what could be done with the pyramid. We had a bit of a breakthrough moment when we realized that we could use the soft-pots to send pitch data and the buttons to control various parameters. We divided the soft-pots into eight equal regions by bracketing the data we received from them in Max[2]. The number eight was chosen because it allowed a full major or minor scale (or mode), and still left the regions for each note large enough to accurately target the desired note. Further, we had the idea to add force-sensing resistors (FSRs) beneath the soft-pots in order to control relevant parameters with pressure while a note is being held down. This created a very natural feeling way to control parameters such as vibrato and wah or other filter effects.

Because the FSRs that we had access to were square and smaller than the soft-pots, we created a 3D-printed hard interface[3] that allowed pressure to be transferred relatively equally from any part of the soft-pot to the FSR. This interface was designed as a disc with eight circular holes in it corresponding to the eight note-regions around the face of the soft-pot. These holes function as a tactile reference when playing, improving ease of performance and preventing disorientation playing the instrument that could arise from playing notes on a continuous surface. Beneath this interface we put a layer of soft neoprene foam to allow a natural travel when pressing the FSR and make it easier to achieve the desired pressure value.

Circuitry and Construction

The circuitry of the device was all handled on one soldered board. Data from the various buttons and sensors on the instrument was fed into an Arduino Mega unit which sent the data over USB to the computer. The circuitry as a whole was relatively simple, using standard circuits for each component.

The pyramid itself was made of three identical sheets of laser-cut acrylic, affixed to one another with angle brackets, and attached to a triangular wood base via hinges on one side and a magnet on the opposite vertex. The wood base was added to the pyramid when we realized that there would not be enough space for the wiring and electronics without it. The whole thing was spray painted with somewhat hideous primary colors because… why not?

---

[2] See Figure 2
[3] See Figure 3

Reading and Interpreting the Data

The data from the sensors was conditioned and interpreted in Max. We got the data to Max using the Arduino2Max patch provided to us. The FSR and soft-pot data were used to generate note-on, pitch, velocity, and aftertouch control data using the subpatch shown in Figure 4. The data from the buttons of each face were fed into a 1x5 matrix which generated a binary value from 0-31 for each unique combination of buttons being pressed. This was then converted to a plain integer which could be used to trigger one of 32 different patches which could do anything from changing the timbre of a sound, to adding various jftypes of polyphony, changing the scale of the notes in the circles, switching octaves, etc. Unfortunately, there was not enough time to actually implement this complex feature, so instead for our presentation the buttons acted independently, rather than in concert with each other, only triggering one sample or function per button.

Figure 1:

Figure 2:

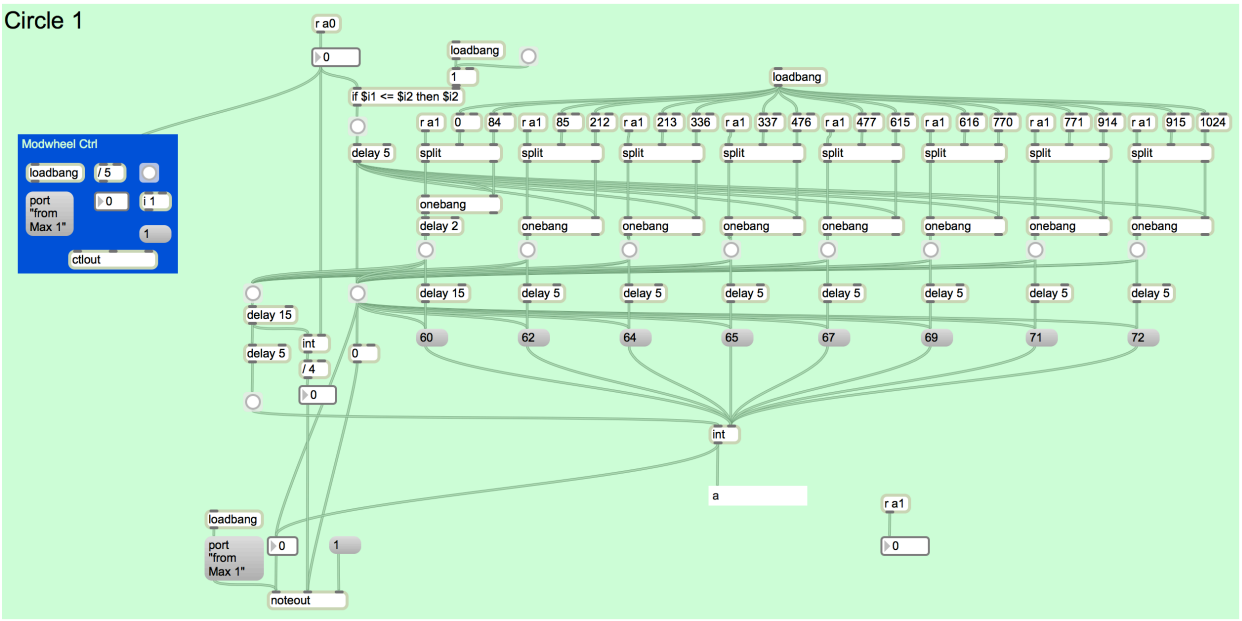


Figure 3:

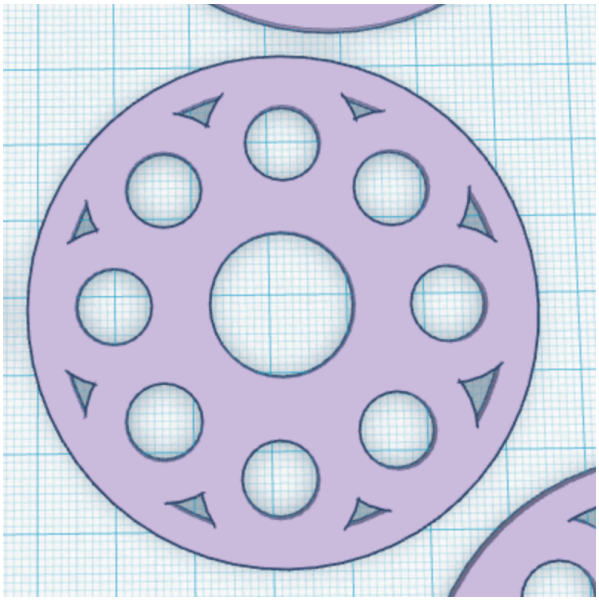
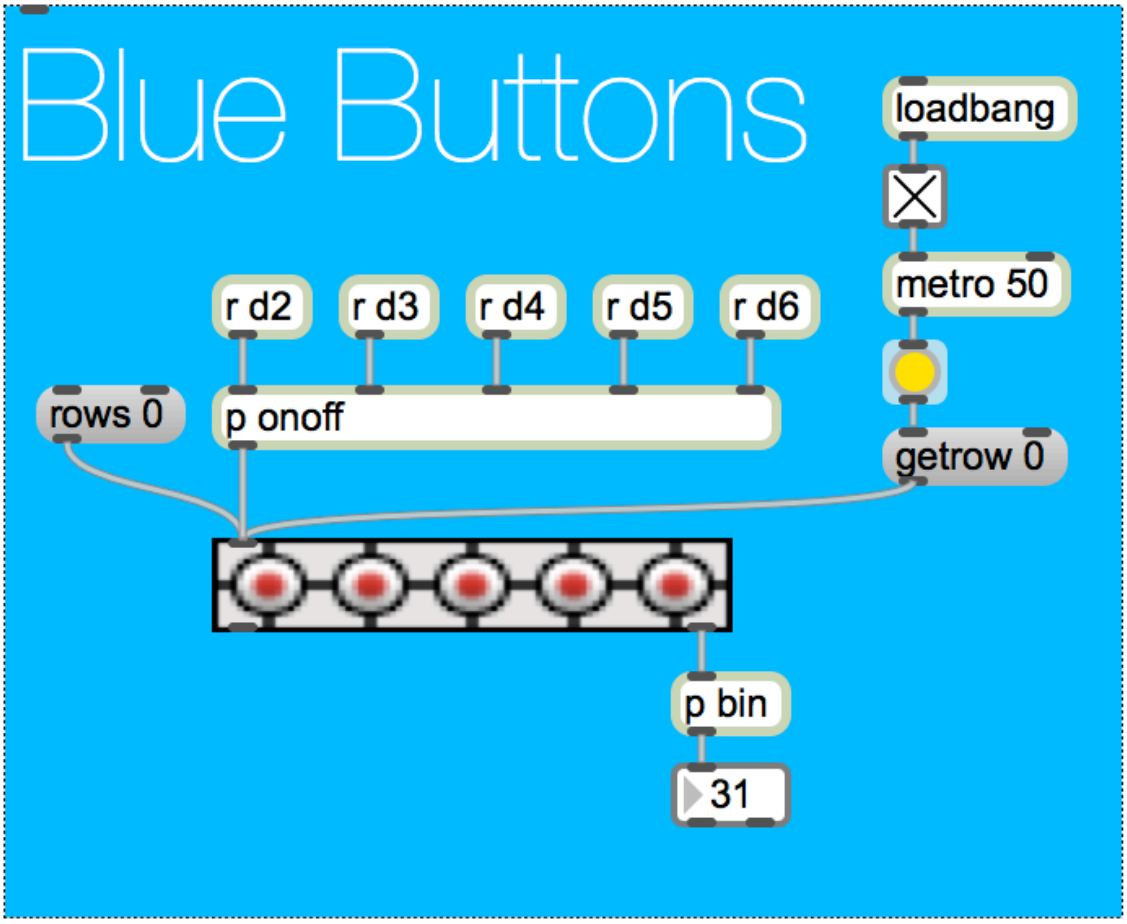


Figure 4:


Blue Buttons
loadbang
metro 50
r d2
r d3
r d4
r d5
r d6
rows 0
p onoff
getrow 0
p bin
31