AXPendula

Alex Gourlay

1 Introduction

For the AXPendula installation, the team set out to provide an interesting audio/visual display that could be enjoyed by multiple people in a room and would incorporate some level of interaction.

The resulting display of eight large swaying pendulum, each playing a note as they passed through their centre points and illuminating in accordance satisfied all of these goals. The resulting experience was said to be 'hypnotic' and 'calming' by some viewers.

The installation was achieved by using an Arduino as an input/output controller and MAX msp for processing and sound generation, along with software MIRA to display a user interface wirelessly on an iPad.

The pendulum periods and lengths were calculated carefully so to generate an interesting pattern and melody as each of the pendulums came in and out of phase with each other.

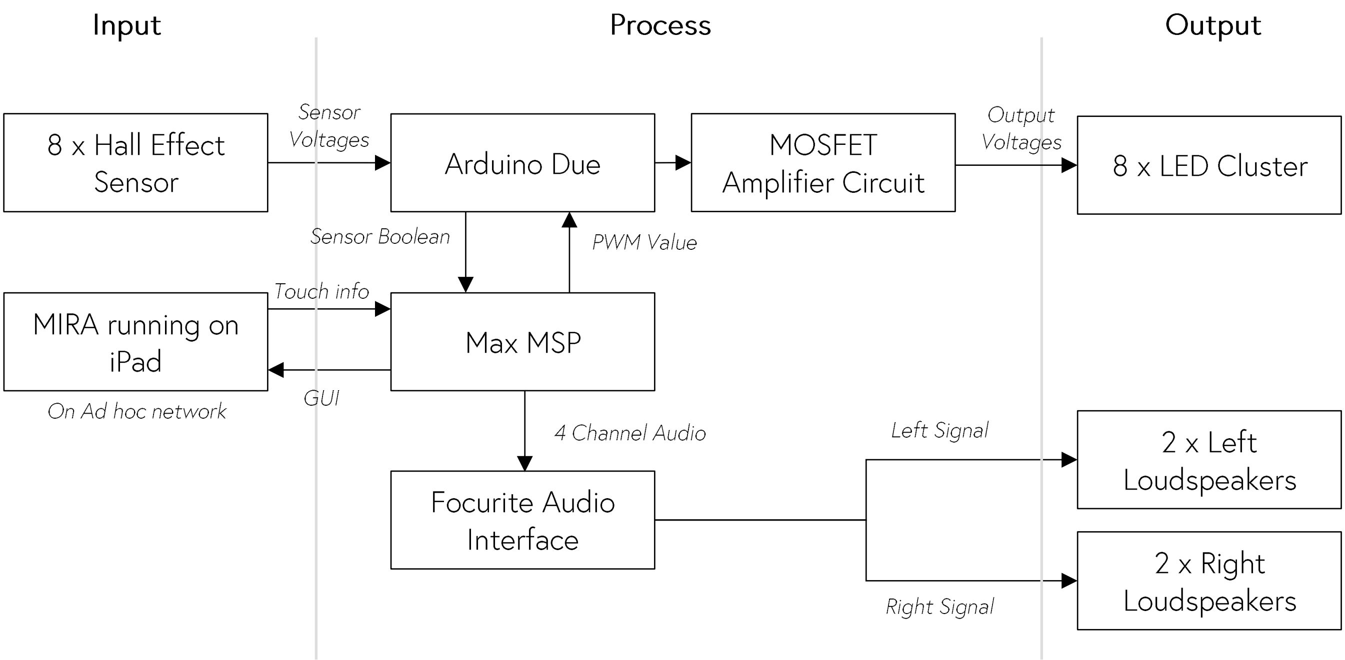
2 Contents

3 Initial Goal and Work Plan

The initial goal was to create eight swaying pendula, constructed from ceiling height that would trigger a note on scale and illuminate when they passed through the centre point. The work was initially broken down into 3 work packages:

1. Working out light intensity, diffusion and connections
    a. Designing the LED bulbs
    b. Connecting lights to Arduino
    c. Controlling light intensity through MAX msp
2. Constructing the pendulum
    a. Calculating lengths
    b. Building the structure
    c. Working out a release mechanism
3. Hall Effect Sensors and sound creation
    a. Routing the sensor inputs to Max MSP via Arduino
    b. Creating a sound in MAX msp, to be triggered by sensors
4. User Interface
    a. Designing a user interface
    b. Implementing using an iPad or other means

4 System Diagram



5 The Build

5.1 Pendulum Calculations

The pendulum lengths were calculated so that all the pendula would return to the same phase after a set amount of time 'T'. Having decided the length of the longest pendulum to be 1.6 m, based on the height of the room, this value was used to find it's period using:

$$Period = 2\pi\sqrt{\frac{l}{g}}$$

Where l = vertical length of pendulum, g = acceleration due to gravity

This period was multiplied by 30 (number of cycles, N) to give a T value of 76 seconds. The lengths of the shorter pendulums were calculated by dividing T by a greater number of cycles, incrementing by 1 extra cycle for each pendulum. This gave their periods which could be used to calculate their lengths.

Table 1 Calculated Pendulum Lengths and Periods

Pendulum	Period /s	N	T /s	l /m
1	2.54	30	76.16	1.600
2	2.46	31	76.16	1.498
3	2.38	32	76.16	1.406
4	2.31	33	76.16	1.322
5	2.24	34	76.16	1.245
6	2.18	35	76.16	1.175
7	2.12	36	76.16	1.111
8	2.06	37	76.16	1.051

5.2 Pendulum Design

The pendulums were suspended using transparent polyamide thread and were constructed from several components:

- A 2W, white LED cluster light
- A transparent, plastic Christmas tree bauble that was sprayed with a matt acrylic coating in order to better diffuse the light
- A ceramic 3D printed holder with an interference fit clasp for the LED cluster, a snap fit for the bauble and a hoop for the suspending thread
- A 1 cm diameter neodymium magnet, adhered to the base of the pendulum

The 3D printed holders were filled with rice in order to increase their inertia when swinging , allowing the pendula to swing wider for longer.

Figure 2 Final pendulum, filled with rice

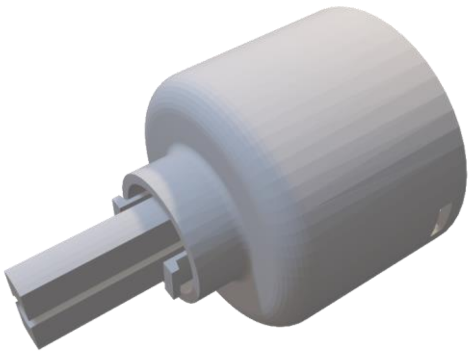


Figure 1 CAD model of pendulum holder

5.3 Hall Effect Sensors

5.3.1 Testing

After purchasing and receiving the sensors, they were tested to see if they worked with the purchased magnets and the Arduino. Using a brief Arduino script, the state of a pin connected to a sensor could be read when the magnet was brought into close proximity:

```
void setup() {
pinMode(A8, INPUT);
}

void loop() {
print(analogRead(A8));
}
```

5.3.2 Supports

In order to raise the HE sensors into close proximity with the pendulum magnets, wooden supports were designed and laser cut. They were painted black fixed to a lower horizontal bar.

Figure 3 Sensor supports

5.4 Loudspeaker Arrangement

The team wanted to implement a method of audio spatialisation for the project and so went about researching ways of doing so. At first the team investigated the possibility of using Ambisonics to accurately spatialize the source of each pendulum's sound. This idea was eventually turned down as the effect would only be convincing for someone stood in a small sweet spot. Instead, the team opted for a linear panning system where the sounds would be spatialized along the length of the instillation.

To do this, a minimum of two loudspeakers was necessary. Since viewers would potentially be stood on either side of the instillation, the team decided to use four speakers as shown below.

Figure 4 Loudspeaker Arrangement

The linear panning implementation is discussed in the [Linear_Pan] section of the report.

6 Max MSP Patch

6.1 Master Patch – [AXPendula]

The master patch, named 'AXPendula', was the hub of the program and consisted of:

- 8 messages, each containing the frequencies of the notes for the chosen scales. All of these messages were routed through an [unpack] object which sent the nth note to the nth pendulum's [Envelope_Sound] sub-patch
- 8 groups of objects; each containing the [Envelope_Sound] sub-patch and the time period for each of the pendulums, along with a [levelmeter~] object for user reference
- The [Linear_Pan] sub-patch that was routed through multiplier objects (which were connected to the level control on the UI) and then into the [dac~] to output the audio through the USB audio interface
- The [maxuino_sub] sub-patch, with eight inputs for the light intensity envelopes and eight outputs for the hall effect sensor states

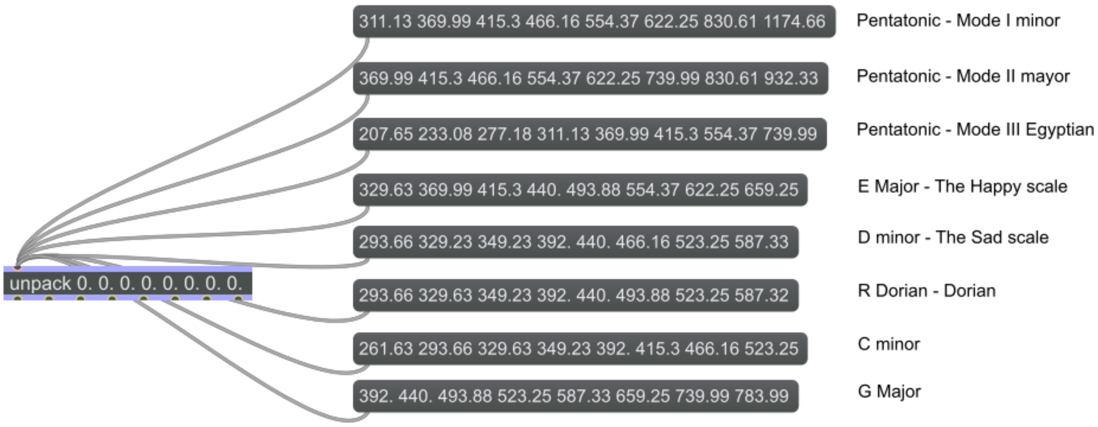


Figure 8 Scale message objects routed through an [unpack] object

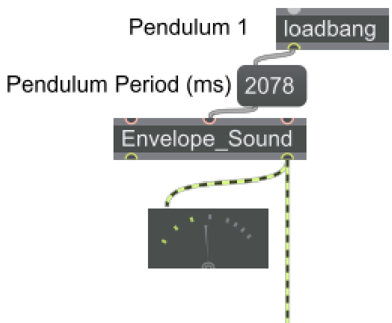


Figure 7 The first pendulum's period message and [Envelope_Sound] object

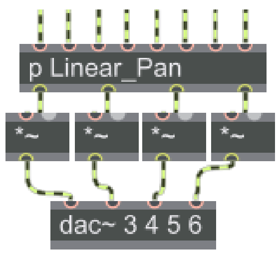


Figure 6 The linear panning object routed to the [dac~]

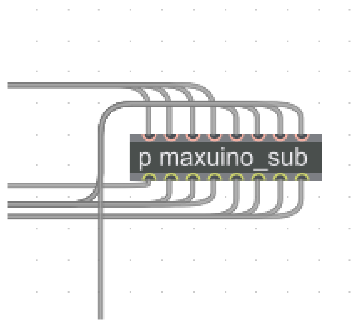


Figure 5 Maxuino object

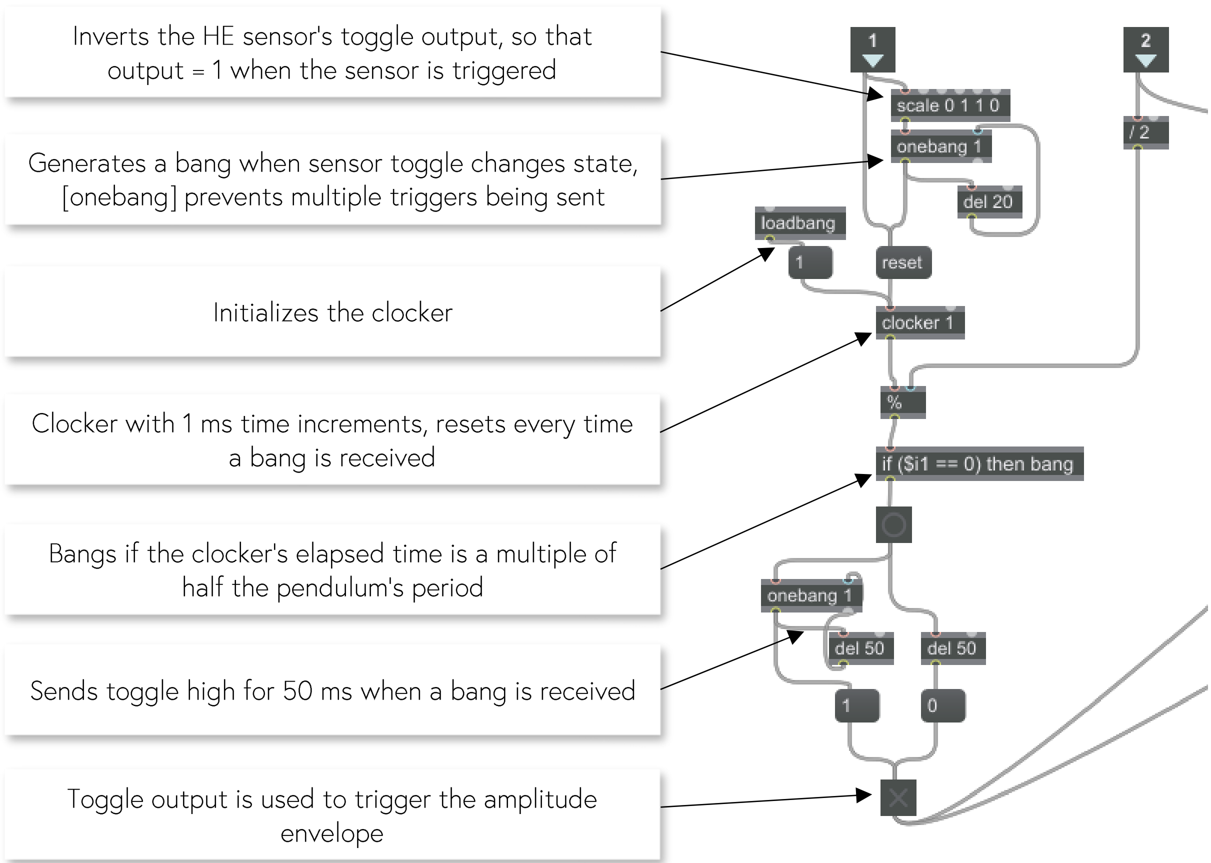
6.2 Sub Patch 1 – [Envelope_Sound]

This sub-patch contained the objects used for triggering and generating the sound/light envelope as well as containing the [sound_generator] sub-patch.

6.2.1 Triggering and Timing

The initial idea for triggering the sound and light when the pendula passed the centre, was to simply trigger the sound/light envelope whenever the sensors changed state. In practice this did not work reliably - for two reasons; the hall effect sensors would sometimes not sense the magnet passing over and the pendulas' trajectories would sometimes not align with the sensors.

A new system was therefore developed using a counter whereby the sound/light envelope would be triggered after every half period, whether or not the sensors had changed state. If the sensors *were* triggered, they would trigger the envelope as well as resetting the counter. This meant that accumulating error from the given period value would be reset, preventing the sound triggering to go out of sync with the position of the pendula.



6.2.2 Envelope Generator

In order to control the volume of the sound and the intensity of the light, Max's [adsr~] envelope generator was used. It was decided that the the envelope should reach peak intensity very quickly before reducing to zero once the pendula had reached max displacement. In order to achieve this, a very short attack time was used (10 ms) and the release time of the envelope was set to a quarter of the pendulum's period. The decay and sustain parameters were unimportant as the evenlope was only being triggered for a very small amount of time (50 ms).

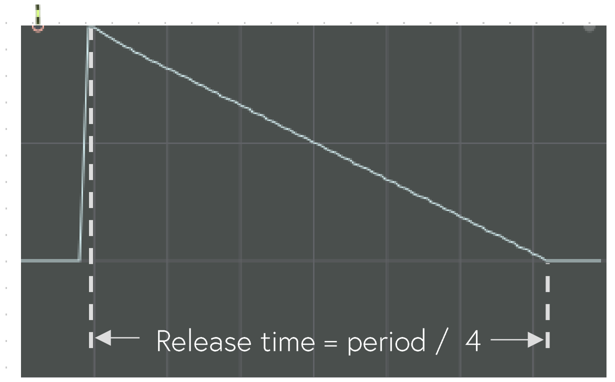


Figure 10 The sound/light envelope, visualised using Max's [scope~] object

The output of the [sound_generator] sub-patch was then multiplied by the envelope signal in order to modulate the sound's amplitude. In order to use the envelope output to control the light intensity (through maxuino), the output had to be converted from a signal data type to a numeric data type. The [snapshot~] object was used to sample the signal at a rate of 100 Hz. Further to this, a value of 0.01 was added to the sampled values to increase the base (untriggered) brightness of the pendulums above zero.

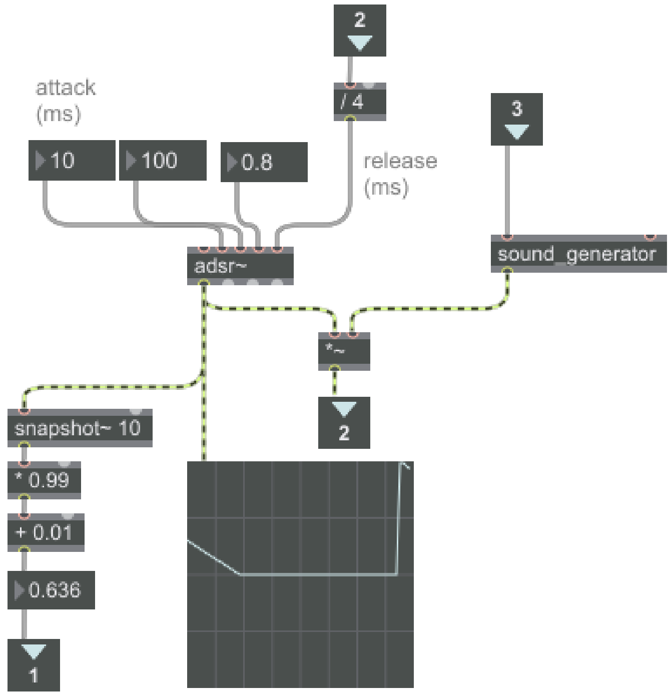


Figure 9 Envelope generator patch

6.3 Sub Patch 2 – [sound_generator]

This sub-patch is where the sound for each of the pendula is generated and shaped. The team wanted a sound that would be congruous with the calm nature of the installation and so created a soft sound using subtractive synthesis.

The sound was generated using a square wave oscillator [rect~] and a sine wave oscillator [cycle~] that was set to the same note 2 octaves lower (achieved by dividing the frequency input by 4) to give the sound better bass content. These oscillators were then summed and passed through a low pass filter [svf~] with a cutoff of 200 Hz and a Q setting of 0.4 to provide resonance. Another [adsr~] object was used to generate an envelope to open and close the cutoff of the filter to 400 Hz , this envelope was set with a very fast attack time (10 ms) and a short decay time (100 ms) with no sustain or release. Using this envelope allowed a very short 'pluck' of high frequencies through, which added sonic complexity to the sound.

The sound was finally passed through an algorithmic reverb [rev3~] object which had been downloaded from an online source, this again added sonic complexity.

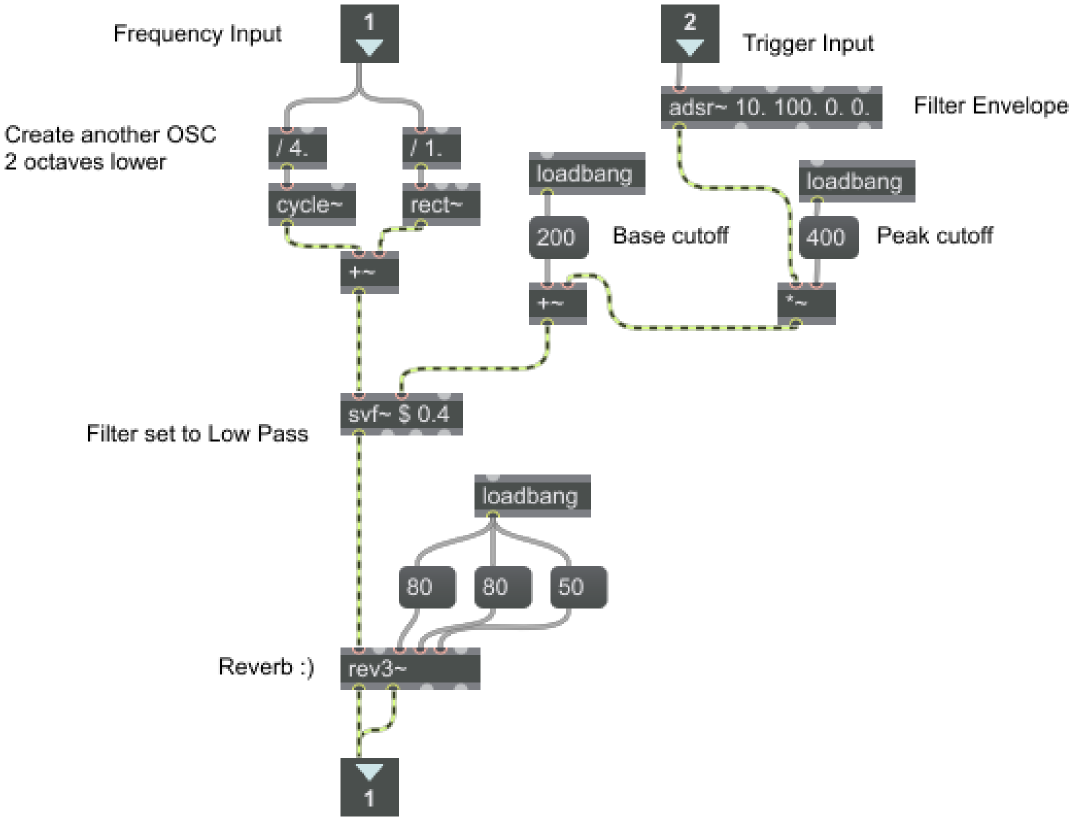


Figure 11 The entire [sound_generator] patch

6.4 Sub Patch 3 – [Linear_Pan]

This sub-patch was used to implement the linear panning used for horizontal spatialisation along the length of the installation, as discussed in loudspeaker arrangement. It operates based on the formulae for linear interpolation:

$$Left\ Speakers = s \cdot \left(\frac{b}{a+b}\right),\ Right\ Speakers = s \cdot \left(\frac{a}{a+b}\right)$$

Where; s = signal, a = distance from left loudspeakers, b = distance from right loudspeakers

Each of the pendula's audio was panned individually using the appropriate a and b values, which were simplified as values relative to the overall length of the installation.

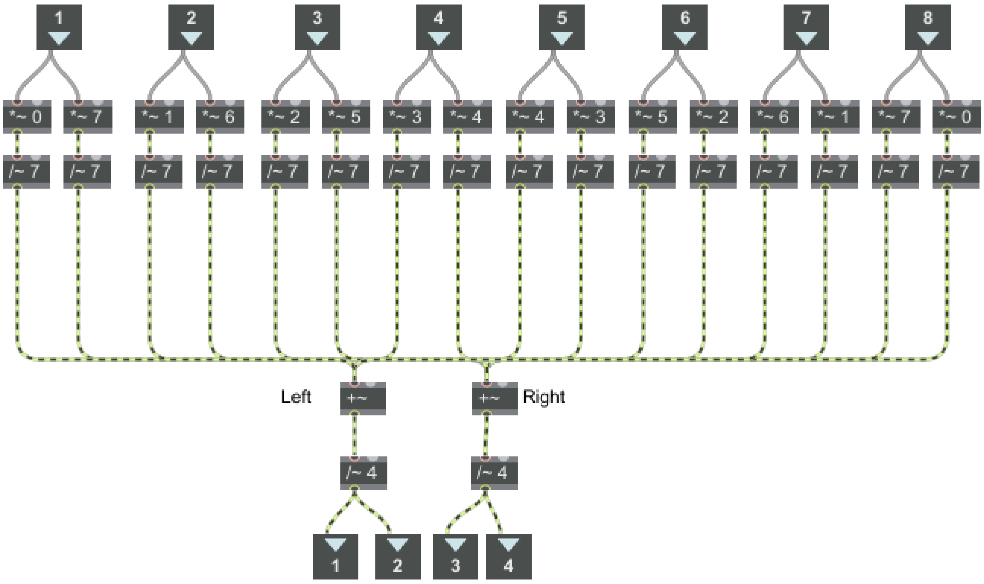


Figure 12 The entire linear panning patch

6.5 Sub Patch 4 – [maxuino_sub]

This sub-patch was adapted from the stock Maxuino patch that contains preassembled patches for interfacing with Arduino, downloaded from an online source. After installing Firmata onto the Arduino (a protocol for communicating with microcontrollers), the Maxuino object was initialized by prepending the 'COM' port that the Arduino was connected to.

The Maxuino GUI, seen below, was then used to set the pin modes – PWM outs for the pendulum LED clusters and digital ins for the hall effect sensors.

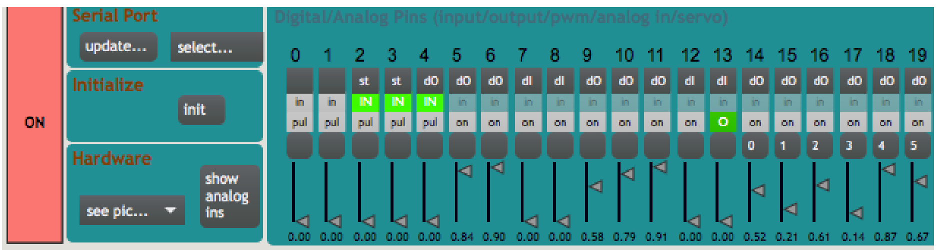


Figure 13 Maxuino I/O GUI

The state of the hall effect sensors could then be acquired by sending the master [maxuino c] object's output through two [route] objects, used to only pass through messages that contained the specified syntax – in this case 'digital' messages.

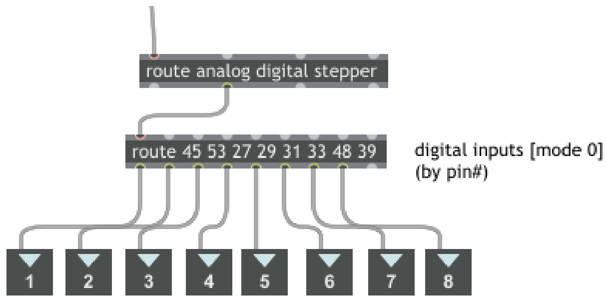


Figure 14 Hall effect sensor digital in state reading

A similar process was used to write to the PWM outputs in order to control the intensity of the

LED clusters, whereby the intensity values (from [Envelope_Sound]) were assigned to the respective pins.

7 User Interface

Having made the decision to include a UI to control the scale that was being played, the team had to decide what scales would be chosen, how to display them and how to set up the UI for use in the installation

7.1 Scales

The team initially thought of developing a function whereby the user could choose from a wide number of different scales by selecting a root note and then a scale mode e.g. 'A' and 'Dorian'. However a simpler approach was ultimately adopted, since it was assessed that a large variety would not necessarily improve the experience. Eight scales were chosen in advance based on the team's preferences, some of which had associated nicknames.

7.2 Implementation

The app MIRA was used to display the interface on an iPad wirelessly over an ad hoc network. By loading a [mira.frame] object into the AXPendula patch, anything moved inside that object frame was displayed and was interactable via the iPad.

The interface graphic was moved inside the [mira.frame] and was underlaid with eight buttons, used to bang the messages containing the scale frequencies. A level meter was also included and was connected so to control the master volume of the installation.

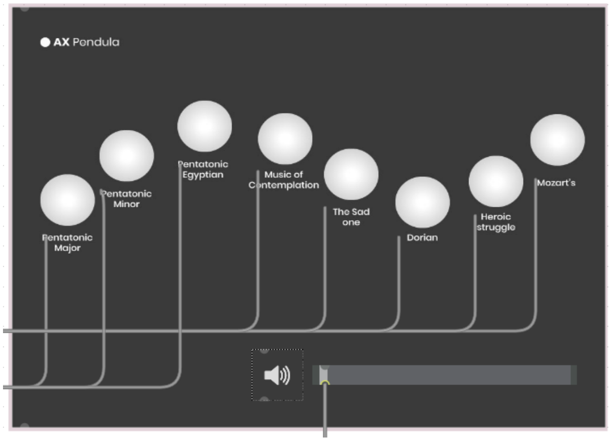


Figure 15 The AXPendula Interface object, patch cables shown

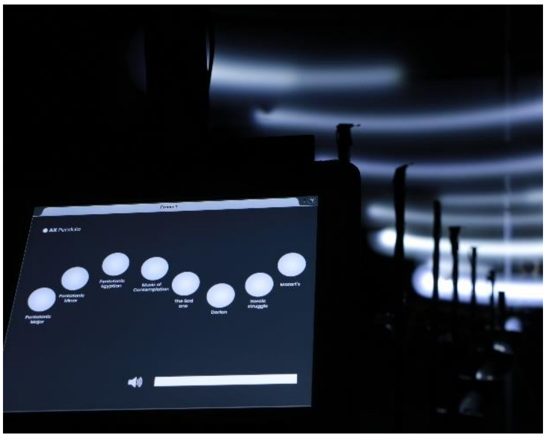


Figure 16 The AXPendula interface in operation

8 Review and Potential Future Work

Overall, the installation was delivered successfully. The display was enjoyed by many people, evident by the number of photos and videos taken and shared on social media. Going forward, some things could have be included or improved.

One thing that could have been included was the ability for a user to record and automatically email the audio file to themselves through the interface. This was attempted during the project by using IFTTT to automatically send the file when it was saved to a Google Drive folder under the file name of the input email - but failed as the saved file name included the file extension.

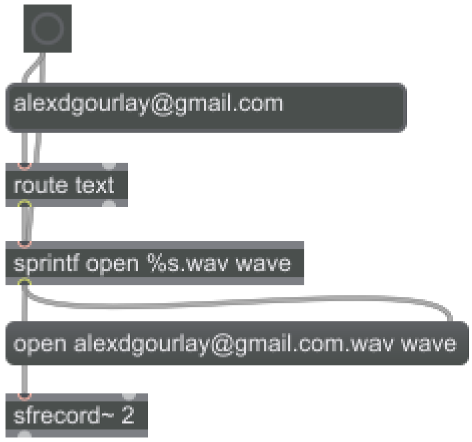


Figure 17 Record and save under email patch

Future work could also involve redoing the project with a larger budget and on a larger scale. Finding a more sophisticated method of initially releasing the pendula (instead of using a plank of wood) would be a first improvement – one potential solution would be to attach magnets to the side of the pendula and to use electro magnets to hold them in a state of displacement before releasing.