The RGB Tonearm: an Interactive Colour Filtering System for Vinyl Records.

Liam Maher
Trinity College, Dublin
Music and Media Technologies
Dept. of Electronic and Electrical Engineering
maherli@tcd.ie

ABSTRACT

The RGB tonearm is an attempt to use the RGB values from a colour sensor in order to interactively filter an audio input. The end goal was to have it mounted on a tonearm so an audio signal from a picture disc vinyl record (or some kind of coloured vinyl) could be filtered live when the sensor was mounted on the tonearm. The values from this sensor are sent to an Arduino board and then sent to the Processing programming environment, where they are mapped to various filters' cut-off frequencies via the Minim library. The user can then choose between four options: a low pass filter, a high-pass filter, or a bandpass filter. There is also the option to listen to the raw audio so that it can be compared to its filtered counterparts.

Keywords: Arduino, Processing, Colour Sensor, Vinyl Records, Interactive Audio Filter Design, Minim.

INTRODUCTION

The inspiration for this project came from the author's experiences DJing and collecting records, and the interactivity that comes from playing with physical records as opposed to digital copies of them. Picture disc pressings have often been maligned in record collecting circles because of the their poor audio quality. This is due to the layer of film that needs to be placed on top of the vinyl for the picture to printed onto. For DJs, the inability to clearly distinguish the grooves that indicate when one song ends and another begins is clearly disadvantageous. However, the vast variety of record pressings are aesthetically very pleasing, offering everything from the aforementioned picture discs, to marble record pressings, as well as simple block colour records. This project re-appropriates these objects and utilises them for the development of an interactive filtering system that offers visual, aural and tactile feedback.

Fig. 1 - Picture and Marble Disc Records

The system itself can be broken down into three distinct stages: the input, processing and response stages. The first consists of sensor input into an Arduino board, while the processing stage consists of the mapping of the sensor input to filters in Processing and then patching these filters to the input audio signal. The response stage is then the filtered audio which is output with its corresponding visual content.

ARDUINO

An Adafruit TCS34725 sensor was chosen for communicating the RGB values of the vinyl records to the Arduino Uno board via the use of the 12C serial bus [1]. The sensor can relay information to the Arduino via the use of its SDA (serial data) and SCL (serial clock) ports which are connected to two of the analog input pins (A4 and A5) of the Arduino board. When the circuit is completed with the connection of the sensor to the Arduino board's 5V and Ground ports, the sensor will emit light from its LED. At this point, the sensor can communicate to the serial bus of the Arduino software via the use of its accompanying library from Adafruit. The lighting environment has a big impact on the sensor itself so an integration time of the data reading needs to be specified in order to suit the environment that it is in. This can be done in the Arduino sketch itself. This sensor was then wired to a break-out board so that it could be effectively placed on the tonearm of the record and so filter the record as it progressed. See Fig. 3.

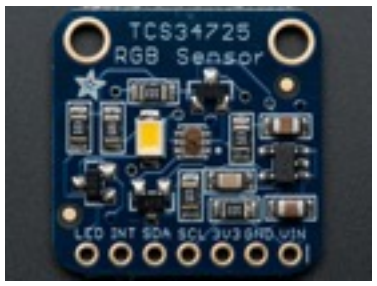


Fig 2 - Adafruit TCS34725 RGB Sensor

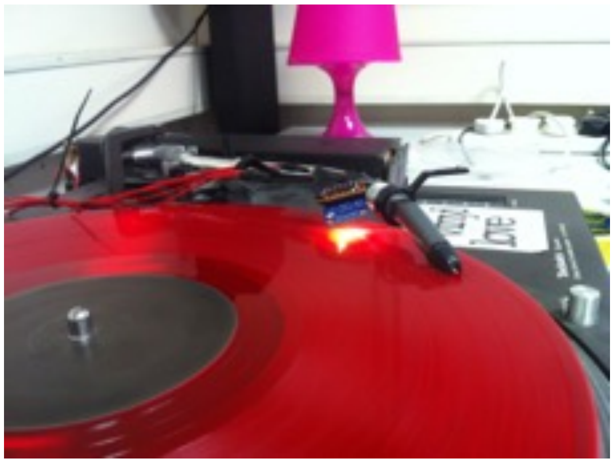
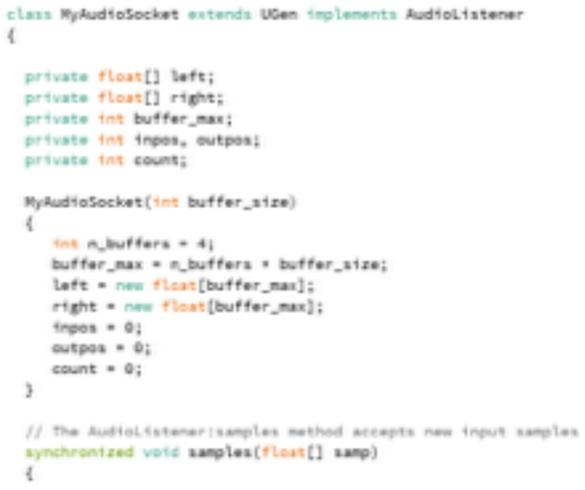


Fig 3: Mounted Colour Sensor

PROCESSING

The Processing sketch takes in the values that arrive from the serial port of the Arduino board and reads them into a String array. This array can then be accessed and split so that the sensor values can be used to map filtering parameters such as the cutoff frequency. The filters themselves were designed using the Minim library for Processing, which offers the user a vast array of synthesis, sound design, and musical properties which can be manipulated using the Java programming language [2]. A routine was then created so that the Minim Ugen routine could be extended and allow 'AudioListener' to listen to incoming audio from the turntable, making real time filtering possible. This is not possible in Minim usually, as different Ugens are sometimes incompatible and as a result, it is sometimes not possible to patch a routine from one Ugen to another. This has been called the 'AudioSocket' class, and allows the required patching for this project. This was adopted from the code found here [3].



```
class MyAudioSocket extends UGen implements AudioListener
{

  private float[] left;
  private float[] right;
  private int buffer_max;
  private int inpos, outpos;
  private int count;

  MyAudioSocket(int buffer_size)
  {
    int n_buffers = 4;
    buffer_max = n_buffers * buffer_size;
    left = new float[buffer_max];
    right = new float[buffer_max];
    inpos = 0;
    outpos = 0;
    count = 0;
  }

  // The AudioListener:samples method accepts new input samples
  synchronized void samples(float[] samp)
  {
```

Fig 3 - Example of Audio Socket Class Creation

This was deemed the most suitable way to present the system given the instant feedback that it provides to the user. It also allows the user to visually track the filtering operations in real-time as the colour sensor's position is clearly presented and accompanied with an on-screen visualisation of the colour that it sees. The filtering operations were implemented using the low pass, high pass and band pass filters from the Minim library for Processing. Their cutoff rates were mapped to values from the colour sensor which was read into a string array in Processing. This could then be split into its RGB values and mapped to the filter's cutoff controls. KeyPressed() functions allow the user to switch between various filter types as they desire. The draw loop of the sketch represents the data that the colour sensor sees by drawing the sensor's input colour as the background. The waveforms of the incoming audio are also represented. The bandpass of the filter is drawn as a white rectangle which becomes bigger as the width of the pass band increases in accordance with the colour values that are mapped to this parameter.

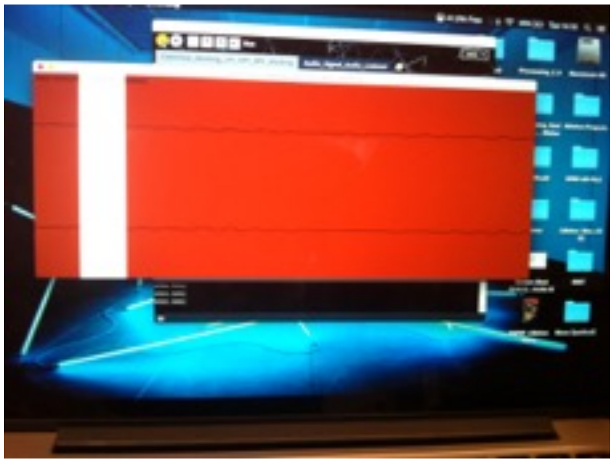


Figure 4: Visual display in Processing with the band pass filter's bandwidth represented by the white rectangle.

HARDWARE

The hardware components of this project consisted of the turntable itself, whose platter and tonearm were exploited for the realisation of the project. The audio of each record was fed into a DJ mixer which grounded the turntable before the audio signal was sent through a sound card and output by the computer. The colour sensor was attached to the tonearm of the turntable so that it could read in colour values from the records themselves and perform the intended live filtering. Finally, a variety of records were used in order to have various samples of different colours and record type (marble, picture and block colour) to test the efficacy of the system. The Arduino board was housed in a black box to the right of the turntable in order to tidy away unnecessary wiring and make the display of the system neater.

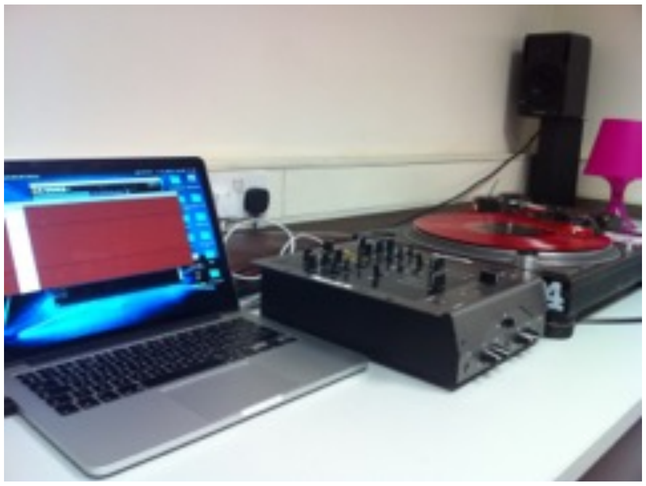


Figure 5: Entire System

ANALYSIS OF SYSTEM

The colour sensor and filters work very well in tandem with each other and the system performs nicely on this front, with no noticeable latency. A constant colour value is always output and the filter mappings work without any problems such as stepping. Importantly, the filtering operations are audible and the graphical feedback for the passband filter is useful as there is a good correspondence between the visual display and the aural output. The use of actual records also offers a more tangible interaction with the system, but it could be useful to integrate the playback of digital files in a manner that will be outlined in the last section of the paper. The user interface could also be further developed as will be outlined, in order to further develop the tangible interface that has been attempted with the inclusion of the hardware components of the project. The turntable offers the additional functionality of being able to manipulate the speed at which the platter rotates, and hence the pitch of playback. This operation is something that can be troublesome when attempting to integrate hardware equipment at the same time.

CONCLUSION: FUTURE DEVELOPMENTS

Some additional functionality such as altering the rate of playback of digital files as well as that of the records could be a fruitful area of research and lead to the further expansion of this project. This could be done using a camera which is equipped with a high framerate and the use of something such as the reacTIVision library for example [4]. This would allow the tracking of playback speed by the use of a fiducial marker, which could then be mapped to the playback speed of a digital file for example via the use of the TUIO protocol. An installation of Jimmy Eadie's called *Wow & Flutter*, could also be further expanded with some of the strategies gained from completing this project in order to extend its initial scope, and make for a more interactive experience for the listener/viewer [5]. In this installation, Eadie exploits the mechanical instabilities of turntables which can cause pitch variations and beating effects on playback. For the installation, eight acetate records were pressed with the same music on each. Acetate records disintegrate more quickly than regular vinyl so the installation itself 'ages' each time it plays. The surface noise of the records themselves after each airing and so, this noise ends up becoming intertwined with (and a part of) the music. If records were pressed up with colours that vary as the record is played, then a dynamic filtering effect could be heard that would go in and out of sync with the other records. Varying filter patterns could also be achieved and spectral splitting of the installation could be achieved. Further developments could include the mapping of different effects in a similar method so real time audio processing could be done via colour. A more elaborate user interface could also be devised for future presentations: for example, instead of utilising keyPressed() functions, a system of buttons could be utilised which would allow the user to cycle through different types of filters, which would provide a more immediate interface for interacting with the system.

REFERENCES

1.Adafruit Color Sensor, 2014, Bill Earl. Available from: https://learn.adafruit.com/downloads/pdf/adafruit-colour-sensors.pdf [04/01/16]

2. Minim Library references. Available from: http://code.-compartmental.net/minim/ [03/01/2016]

3.https://www.ee.columbia.edu/~dpwe/resources/Process-ing/AudioSocketExample.pde

4.ReacTIVision library. Available from: http://reactivision.-sourceforge.net/

5.Eadie, J. *Wow and Flutter*, installation [2014].