# Design and Evaluation Report

**Luke Hammond**
**Digital Media**
University of the West of England
Coldharbour Lane
Bristol, UK
luke2.hammond@live.uwe.ac.uk

submitted 27/04/2022

## Abstract
This report summarizes research in the area of Creative Audio-Visual Application. This contains one webpage which utilizes the `switch()` function to display two states, the code for the images as well as a demonstration video presentation.

## Introduction
I have been thinking about what to do about this project since it was first announced early in the year and after looking at examples of other students during lectures, I decided to take an approach focused on images showcasing amplitude.

Since starting coding this year, I feel confident in approaching a piece of code and knowing what is going on, or possibly fixing it, knowing this I took this assignment as an opportunity to hone my coding skills.

My research method contained looking at tutorials on YouTube, especially those made by 'The Coding Train' as well as looking into code on w3schools and the p5.js reference page.

## Research & Method
My first approach to my work was putting the knowledge I gained from last assignment to use,

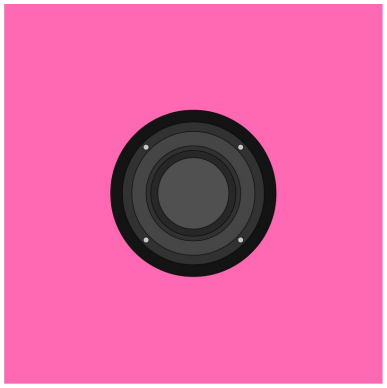

**Figure 1.** Initial design for Speaker.

**Figure 2.** Initial design for Vinyl.

starting with simple drawings for my work which I would then develop up. From here I added the `switch()` case to allow for the two states that the assignment brief states, for this I got help from Dr Dave Meckin so my music that I had selected also started along with the `switch()` case.

```
function keyPressed() {
  userStartAudio();
  switch (key) {
    case "1":
      currentSong.stop();
      currentSong = song1;
      currentSong.play();

      break;

    case "2":
      currentSong.stop();
      currentSong = song2;
      currentSong.play();

      break;
    default:
      currentSong.stop();
      break;
  }
```

**Figure 3.** `switch()` case that allowed for my songs to play on a specific key.

Once I had the basic layout of my work completed, I moved onto my secondary research which focused mainly on YouTube videos and double-checking my understanding of my methods/research via the p5.js reference page as well as worksheets designed by Dr Simon Emberton and Dr Dave Meckin. I used these research methods as I knew I could find what I wanted or even more inspiration for my work.

*Visually showcasing amplitude*

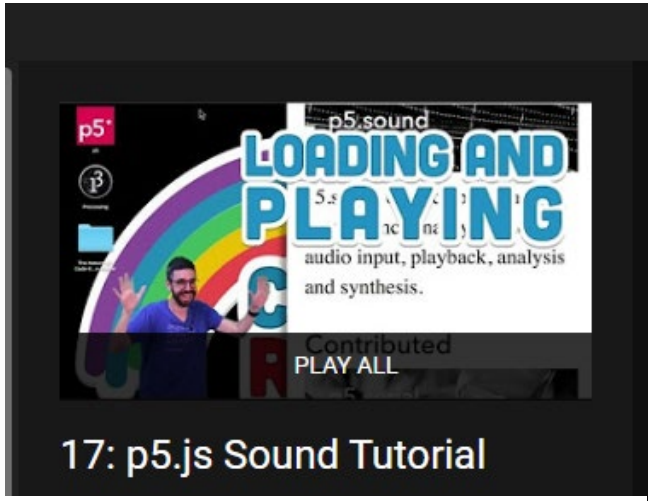


**Figure 4.** Playlist on YouTube by The Coding Train focused on sound in p5.js.

For a lot of my inspiration, I followed Dan Shiffman *'The Coding Train'* on how to manipulate my shapes into moving along with the amplitude. In these videos, he mentioned `p5.Amplitude()` as well `.setVolume()`. Normally, amplitude readings can only go from 0-1, however with `setVolume()` this meant the reading of the amplitude wouldn't go higher than the designated volume.

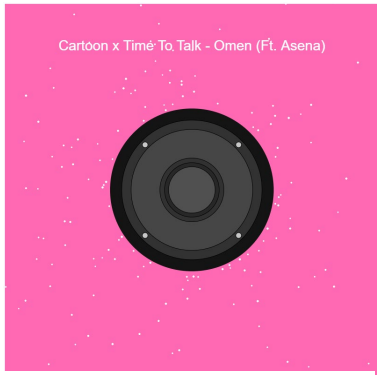


**Figure 6.** Finished product of Speaker.

*Speaker research*

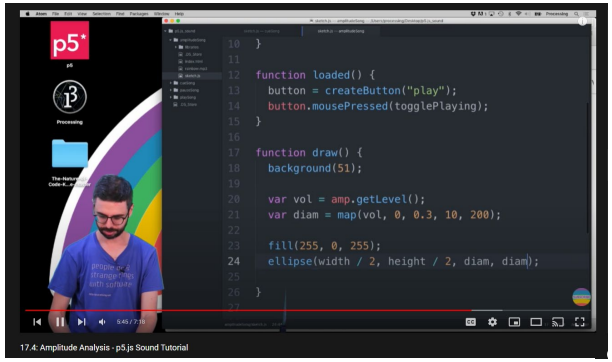


**Figure 5.** The Coding Train's video about Amplitude.

This video allowed for me to change the width of my selected `ellipse()` in my Speaker drawing. They expand from a starting diameter size to a max diameter size, the change in size then corresponds to the reading of the amplitude at the time, the higher the amplitude reading the closer the `ellipse()` gets to reaching its max expansion size.

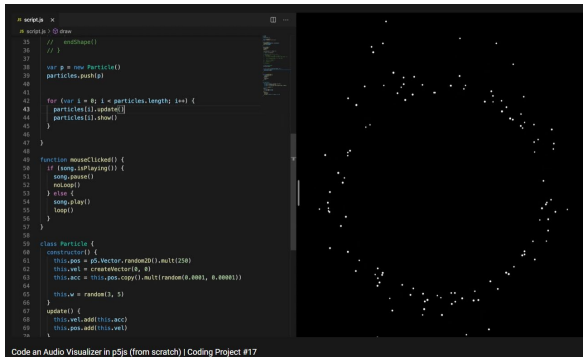


**Figure 7.** Colorful Coding's video about particles moving to amplitude.

This video was the final step to finishing my speaker visual state, it showcased particles that move in response to the amplitude. Watching this video taught me two new pieces of code; `random2D()` and `mult()`. These were crucial in getting the particles to appear randomly and on mass. `random2D()` "makes a new 2D unit vector from a random angle" (p5.js reference page) and `mult()` is used for multiplying vectors, meaning together `random2D()` will spawn the vectors in and `mult()` will increase the amount being called.

In addition to this, the video further helped me understand other code I was slightly confused on such as `.splice()`- I managed to grasp the idea of inserting more data into an array.

*Adding text*

During the time I was completing worksheets I discovered a simple way of centring text. Until this moment in time, I was roughly estimating where the centre of the screen pixel by pixel. However, I soon discovered that I could simple `textAlign(CENTER)` my text and set my text position to `(width/2)` and then the y-axis as I desired.

```
//Setting the size in px, colour and position of the text
push();
fill(255);
textSize(30);
textAlign(CENTER);
text("Cartoon x Time To Talk - Omen (Ft. Asena)", width/2, 100);
pop();
```

**Figure 8.** Centring text on a canvas.


**Figure 9.** Inspiration for the Vinyl state.

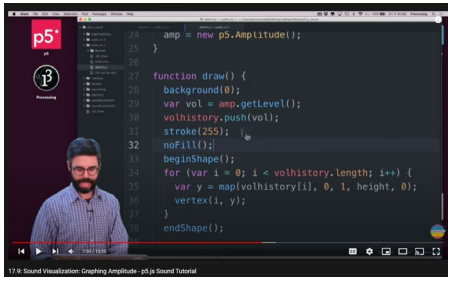


**Figure 11.** Video by 'The Coding Train' about graphing amplitude.

*Vinyl research*

To give my work more of an authentic feel I had to recall how to rotate my work, this first involved refreshing my mind.

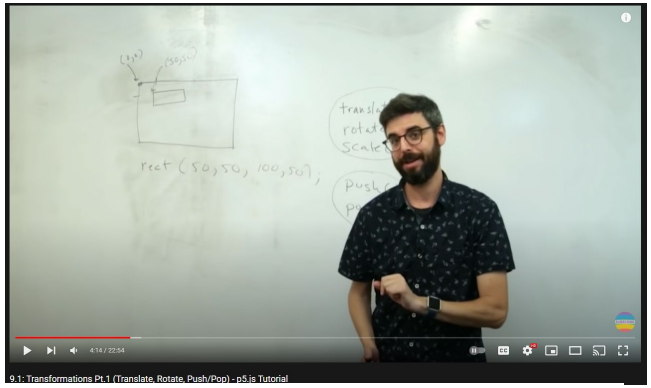


**Figure 10.** The Coding Train's video explaining `rotate()`, `translate()`, `push()` and `pop()`.

Following this explanation, I got my vinyl record to rotate. However, it was not very clear that it was spinning, to make it more apparent to the user I simply added a face to showcase the rotation as seen in Figure 2.

To achieve similar results as to what is shown in Figure 9, I first needed to get a visual representation on my amplitude. Following a Dan Shiffman video (Figure 11), I laid the foundation for my desired outcome.


**Figure 12.** The Coding Train's video explaining radial graphs.

The video in Figure 12 helped in aiding my graphed amplitude to circle my vinyl record through the use of `map()`. The `map()` function maps designated points, this allowed me to set a diameter similar to my vinyl record and the max diameter the amplitude reading can span out to.

The circle was created inside of a `for()` loop which stated "`i`" could not go higher than `360` - a circle. To create the circle I had to make use of the `cos()` and `sin()` function which designated that the mapped graph would be a circle.

```
for (let i = 0; i < 360; i++){
  let r = map(volhistory[i], 0, 1, 125, diam);
  //Creating a circle
  let x = r * cos(i);
  let y = r * sin(i)
```

**Figure 13.** Code to create a radial graph.


**Figure 14.** Final version of the Vinyl state.

In Figure 13 there is the inclusion of `diam`, this causes the radial graph to increase in size corresponding to the amplitude, this follows the same code as mentioned earlier in the *Speaker research,* it follows the video shown in Figure 5. This occurs because the last parameter is changed to `diam`, if I inserted a number instead, the radial graph would not scale in size causing the bouncing effect it has, instead it would only scale in size to a set number.

Before there was colour in my final product I simply used `stroke(255)` which drew a white outline to show the radial graph with no colour inside. However, wanting colour I added the `fill()` function to colour the inside of my radial graph. Although this fulfilled my desired outcome, I was not satisfied and thus wanted to add a colour gradient to my radial graph which caused the colour to change overtime.

```
for (let i = 0; i < 10; i++) {


  let r = map(sin(frameCount/2), -1, 1, 50, 220);
  let g = map(i, 0, 0, 50, 100);
  let b = map(cos(frameCount), -1, 1, 20, 180);

  fill(r,g,b);
```

**Figure 15.** Code to cause the colours to change over time.

After using `for()` loops in my code I managed to strengthen my understanding on them and I now feel confident in knowing what they do when looking at them. The first parameter states the starting value, the second parameter states the initial parameters range and thus how many times the code can execute and the final parameter states whether the number will increase or decrease the initial stated number, in this case `i++` increases the initial parameter.

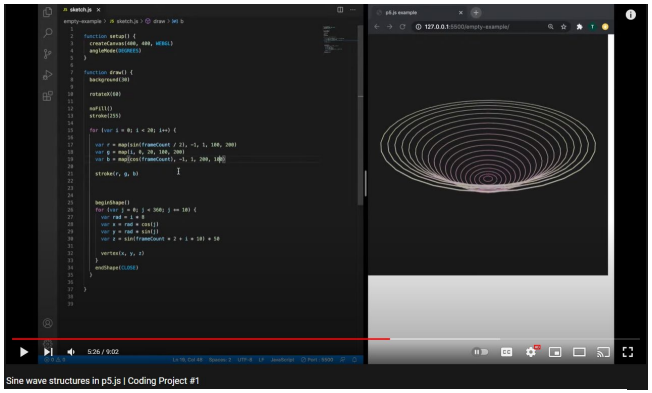


**Figure 16.** Colorful Coding's video about Sine wave structures.

`frameCount()` was important in this loop as it allowed for a gradual fade into another colour, before I added this code found in Colorful Coding's video on "Sine wave structure" my code was changing colour every frame causing the colours to change every frame, which causes a flickering effect.

Figure 15 shows `fill()` is occupied by the variables `r,g,b`, this is what colours the inside of my radial graph and the variables `r, g, b` is what causes the colours to change.

```
//Design on track

fill(255,255,0);
ellipse(0,0,60);
fill(0);
ellipse(10,-10,10);
ellipse(-10,-10,10);
fill(255);
ellipse(10,-10,7);
ellipse(-10,-10,7);
fill(0);
ellipse(10,-10,3);
ellipse(-10,-10,3);
fill(255);
strokeWeight(2);
stroke(0);
ellipse(0,10,15);
angle = angle + 1.25;
}
```

**Figure 17.** The code for the face on the track & the speed the track rotates.


**Figure 18.** Failed state that shows the origin point of the fill.

## Evaluation

*Speaker*

My speaker almost achieved its desired outcome, I wanted to add a background that flashes colour based on the amplitude which would correspond with the particle's movement.

Coding the speaker has taught me a lot about code as well as solidified my understanding on a lot of aspects of code such as classes. Coding the speaker has further developed my knowledge on `for()` loops and `if()` statements. I believe coding this has made me more confident in possibly coding something similar without a tutorial going forward into the future.

*Vinyl*

My vinyl state had quite a few problems which have either been fixed, changed or covered up within the work to make it look more presentable. One noticeable problem, was when I originally used `fill()` and the origin point was outside the vinyl and it made the image look sloppy this can be seen in Figure 18.

I could not find the solution to this, however I found another method which I mentioned in my Research & Methods, instead of fixing the problem it covered up what was occurring, with the amplitude of the song starting low the original filled circle hardly moves away from its origin point.

Originally, there was no face on my vinyl piece but after I added the rotation it was not very clear that the vinyl track was spinning to make it clearer, I added a face to show it spins. Another problem I found, was increasing the second parameter of my `for()` loop effected the frames of my work due to there being more range of colours.

In addition to this I wanted the whole piece to move as one, in time with the amplitude like the speaker, however I couldn't find an efficient way of doing this besides setting multiple variables to set a range of parameters as shown in Figure 19.

```
let diam = map(vol, 0, 0.25, 100, 200);
let diam2 = map(vol, 0, 0.25, 80, 180);
let diam3 = map(vol, 0, 0.25, 70, 150);
```

**Figure 19.** Setting the variable `diam`.

*Overall Code*

One of the main issues I found in my code was moving or removing `push()` and `pop()` as this could effect code in other JavaScript files. This was very apparent with the text not appearing in the centre of some of my canvas's.

My main critique of my work is that when the song resets itself after you press the corresponding key, the image continues from where it once was, instead of reloading the page and resetting itself. I believe this could be a simple fix and would follow the same concept as what happens with the music.

One of the things I tried was implementing 3D to my work by adding `WEBGL` to my `createCanvas()`, this changed the positioning of all my work due to the origin point being in the middle of the screen rather than the

top left. This could be clearly seen in my vinyl state as the code had a `translate()` function.

An improvement I would make going into the future is my ability to make drawings within JavaScript, while my image is clear, I believe it could be more detailed and possibly tribute more towards the work.

## Conclusion

My work this time is a level that truly reflects where I am with coding, if I was to redo this task I would most likely not need tutorials and could perhaps code this without assistance. This assignment has solidified my coding abilities and I hope to improve moving forward especially implementing the things I missed out on doing here in my later works. I believe if someone was to look at my work they could tell the approach I took with my work and following my evaluation could even add the desirables to my work that I could unfortunately not complete at my current ability in coding.

## References


[1] Cartoon (2015) On & On (feat. Daniel Levi). NCS [download] [Accessed 05 April 2022]

[2] Cartoon, Time to Talk (2021) Omen (Ft. Asena). NCS [download] [Accessed 05 April 2022]

[3] Coding, C. / YouTube (2021) Code an Audio Visualizer in p5js (from scratch) | Coding Project #17. Available from: https://www.youtube.com/watch?v=uk96O7N1Yo0 [Accessed 05 April 2022]

[4] Coding, C. / YouTube (2020) Sine wave structures in p5.js | Coding Project #1. Available from: https://www.youtube.com/watch?v=vmhRlDyPHMQ&t=325s [Accessed 11 April 2022]

[5] Meckin, D. / Github (2021) Week 04. Available from: https://github.com/davemeckin/Intro_to_Creative_Programming/tree/master/Week_04 [Accessed 04 April 2022]

[6] Meckin, D. / Github (2022) Week 19. Available from: https://github.com/davemeckin/Intro_to_Creative_Programming/blob/master/Week_19/Week_19.md [Accessed 04 April 2022]

[7] p5.js () Reference. Available from: https://p5js.org/reference/ [Accessed 11 April 2022]

[8] Shiffman, D. / YouTube (2017) 9.1: Transformations Pt.1 (Translate, Rotate, Push/Pop) - p5.js Tutorial. Available from: https://www.youtube.com/watch?v=o9sgjuh-CBM&t=229s [Accessed 22 March 2022]

[9] Shiffman, D. / Youtube (2016) 17.4: Amplitude Analysis - p5.js Sound Tutorial. Available from: https://www.youtube.com/watch?v=NCCHQwNAN6Y [Accessed 05 April 2022]

[10] Shiffman, D. / YouTube (2016) 17.9: Sound Visualization: Graphing Amplitude - p5.js Sound Tutorial. Available from: https://www.youtube.com/watch?v=jEwAMgcCgOA&t=8s [Accessed 05 April 2022]

[11] Shiffman, D. / YouTube (2016) 17.10: Sound Visualization: Radial Graph - p5.js Sound Tutorial. Available from: https://www.youtube.com/watch?v=h_aTgOl9J5I&t=13s [Accessed 05 April 2022]

[12] w3schools () HTML Color Names. Available from: https://www.w3schools.com/colors/colors_names.asp [Accessed 21 March 2022]