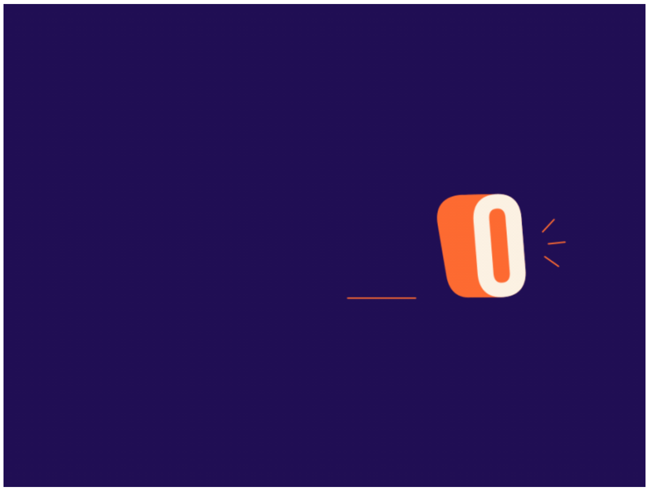
DL836 - CC4 - Generative Design
Assignment 2 - James Blair - N00162202
Generative Type and Audio

Introduction

The application developed for this assignment was an audio visualizer that was implemented on the html canvas using various libraries including p5.sound, p5.gui, opentype and g. The application is an audio visualizer that displays animated effects in synchronization with a music source. The application has a number of user controlled inputs including the primary functionality of the tool, a system by which words are displayed and then transformed into other words.

Type

The primary objective of the application was to create a tool which could automatically transition any type into another. This kind of transition is common in frame by frame animation such as in the below example.



5/57ca855931f0531c9617d0f8ce8fa6ca.gif

The means by which the application achieves this is through the conversion of a font file into it's vector coordinates and storing these through arrays. The multidimensional array for one word is then re-sized and has its points redistributed among its form according to the dimensions of the next subsequent letter it will become.

This algorithm is broken into many parts with separate functions handling multiple sections of the process:

1. **Calculating a new word**

In order to add new words to be created the user types into an input field and adds a word using a button to submit. This word is then added to an array through the 'newWord' function and the points which make up its glyphs are placed into the shapeArray which holds the word currently being drawn to the canvas.

2. **Resizing the arrays**

Once the word is drawn to the screen using lines between the vertices a temporary array is created for the next word in the sequence. The differences in the positions in the two arrays are then normalized and additional points are added in discrete random positions throughout the existing word. Alternatively points are removed from the existing word to accommodate letter forms with fewer points. This process is handled in the method ‘resizeArray’.

3. **Transitioning the word**

The points are moved to their corresponding point in the target array using the lerp function for a smooth transition which slows as it arrives. Once a point reaches its target it is considered arrived. When all points have arrived the transition is considered completed and a boolean flag is lifted to permit the next transition.

4. **Handling extra letters or removing unneeded ones**

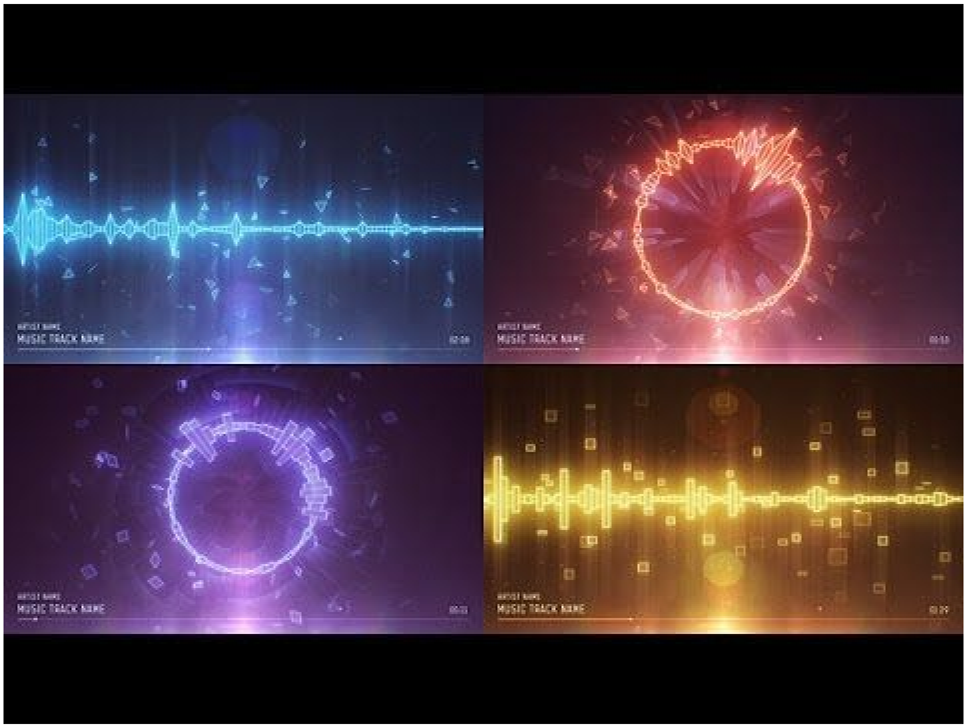
Separate handles were created to calculate the lengths of new words so as not to have letters pop in or out of nothing. Additional letters are animated separately to move to points within the new word or conversely to emerge from existing points. The result is a more natural and consistent animation process without the word length limitation causing jarring motion.

Each of these processes was developed incrementally one after the other so as to maintain functionality throughout while allowing for more effective debugging and testing.

Background Splines and Sound

Once sound files were incorporated using the p5.sound.js library, I extracted an array of all of the amplitudes at each frame and averaged them. When this average is exceeded, The animations occur excluding a central ellipse which is mapped in scale to the exact amplitude per frame.

The background was inspired by more traditional audio visualization software such as the ones in the image below.



Example inspiration at link

The effect was achieved using a large circular series of splines which move towards vertices at the midpoint of their origin to the center and back again at random based on each vertex's assignment of either 0 or 1 on creation. The density of the splines may be dynamically adjusted during runtime as well as the sensitivity at which its animation is triggered.

User Controls

The user interface was implemented using p5.gui in combination with ‘quickOptions’ js library as well as several standard p5.js UI elements such as dropdown menus and input fields. The controls allow modification of the following.

- The sensitivity of on screen elements to amplitude
- Colour of various elements
- Spline density
- Speed of letter morphing animation
- Font weight
- The selected audio track
- The selected font
- The word sequence being displayed as well as the ability to reset the application without refreshing the page.