**Music Visualizer Report**

**1.**

For this project, I decided to implement the music visualizer extensions. My decision was based on the fact that I do like music, and it always fascinated me how the music interfaces were implemented. It was a good opportunity for me to understand and to implement that kind of features. Moreover, I Knew that it had a lot to do with the physics of music and I wanted to understand how trigonometry would be involved and how I can implement real features with that kind of abstract materials.

The first one, is the noise line that acts similar to a random moving line. I would need to combine the noise function and the map function provided by p5.js. By doing so, it is possible to generate x and y coordinate for the vertices. I will program the movement of this line relatively to the energy provided by highMid frequency. Therefore, the line will look like it is dancing to the HighMid frequency. Moreover, the line will change color depending on the energy provided by the HighMid frequency. Since I will follow the RGB color convention, the color of the line will be represented as follow: [(e\*3)%255,255-((e\*3)%255),0], where *e* is the energy. Here, there is the reminder operator to make sure that no color value will exceed 255.

The second extension that I will implement is the ridge plot. The ridge plot consists of a 2-dimensional array and each row represents coordinates for the vertices on the same line. To make the plot, I have to gather the lines in an artificial frame. The lines will move from the bottom to the top with the help of a nested for loop. These loops iterate through the 2D array, and when the top line is displayed, the next iteration will remove that top line to let place for a new line. The new line will be added by a function named addWave(). I plan to make the wave lines change color depending on the energy from a randomly selected frequency type.

The third extension is a circular wave representation. So, the wave line will change upon the value of the amplitude given by wavefrom() method. The circle will be drawn in 2 steps, one for the right half cercle and the second step for the left half of the circle. To do so, I have to change the angle mode to degree and use sin() and cos() function to compute the right coordinates for the vertices.

2.

During the whole development process, I wanted my code to be modular and flexible. Therefore, I made multiple classes and each class perform a certain task. The separation of concerns in this case was very helpful because it allowed me to debug my code very easily, and when a problem occurred it was easily found and targeted. So, each extension has it own class, in instance ridge plot class, noise line class, and disco class for the circle shape. A control class is used to display text and take input from the user. Therefore, at first, it displays the play button and a message:” Press Space for menu options”. By doing that it helps the users to start with the application. Once the space key is pressed the first message disappears and let place for the menu options. The user can choose from option 1 to 3 in order to select a shape to display. The structure also has a sketch class that acts like the main program and do the setup. The playbackButton class is the class that implements the form of the button and can be instantiated to display it on the screen by its draw function. Also, there is a visualization class that contains all the visual objects. So, it let place for upcoming new extensions if needed without changing a lot of code. So, this design approach is very modular and makes the code very readable.

So far, I made very good progress. I have implemented 3 extensions, 2 from the given videos and a custom one. During the process, I would write, first, a high-level solution for each extension by breaking down the problem to subproblems. After that, I would implement each sub problem and test it. At the end, I would integrate everything together and do integration tests to make sure that new added extensions do not break the previous code.

For the upcoming iteration, I plan to make my code cleaner by trying to make my code even more modular if it is possible. After that, ill try to improve the complexity and the creativity of the extensions implemented so far. Then, Ill do more integration tests and unit tests to make sure that my program is bug free.

3.

Graphical user interface, table

Description automatically generated

For the part 1 of the project, I divided the development into multiple steps. There was a preparation phase where I went through the template code and understand the flow. This part was critical because it allowed me to target the part of the code that I had to change and what part can be reusable. There was also a period of documentation where I had to learn more about p5.js and target possible built-in functionalities to use. After that, it was mainly about development and testing, so each time I would implement an extension I would test it.

Table

Description automatically generated

Chart, waterfall chart

Description automatically generated

For the second part of the project, I plan to assess a period to clean the code and find ways to make it even more modular. Once that is done, I will improve the complexity of the extensions, in instance the noise line extension and the ridge plot extension. Indeed, after the implementation, I will test them, and I will do an integration test to make sure that the modifications did not break the app. After that, I will write some documentation to report the work done during the whole process.

4.

During the development of the music visualizer, I used the internet to help me. The most used resource is the p5.js Reference:

* <https://p5js.org/reference/>

I used YouTube to help me with the implementation of my custom extension:

* <https://www.youtube.com/watch?v=uk96O7N1Yo0&ab_channel=ColorfulCoding>
* <https://www.youtube.com/watch?v=MzhBizCmpi8&ab_channel=ColorfulCoding>

Text

Description automatically generated

That is a piece of code pretty similar to what can be found in the videos.