Sam McMahon
WebGL Final Project Report
CS 464
12/5/17

Project Proposal:

For the WebGL Final Programming Project, I'm proposing that it be a music visualizer that modifies the height and color of a terrain at a certain spot based on the Fourier transform of a music file at a certain window in time.

Visually, the screen will contain an initially flat and square 3D terrain. The idea is that the visualization itself will be based off the power of a certain frequency of a song at a certain window of time. Depending on the frequency, that will pick a certain position in the terrain to modify the height.

Depending on the power of a certain frequency of the song at a certain window of time, that will modify both the height of the terrain at the point and the color of it as well. Smaller powers will have the color green and larger powers will have the color red. Powers in between will have colors interpolated between the two.

It's planned that the terrain will be generated in a similar way to the terrain used in both Assignment 2 and Assignment 3. It will be a mesh of triangles, height will be modified by indexing into the geometry buffer at a certain index and modifying the triangle's y coordinate.

To get the data from the music to do the visualization, it's planned that a fast Fourier transform (FFT) will be run on the music in order to get the power over frequency data from the music. The music will be split in to segments 1/60 seconds long, and have the FFT run on each one per each frame, so that the terrain will change each frame.

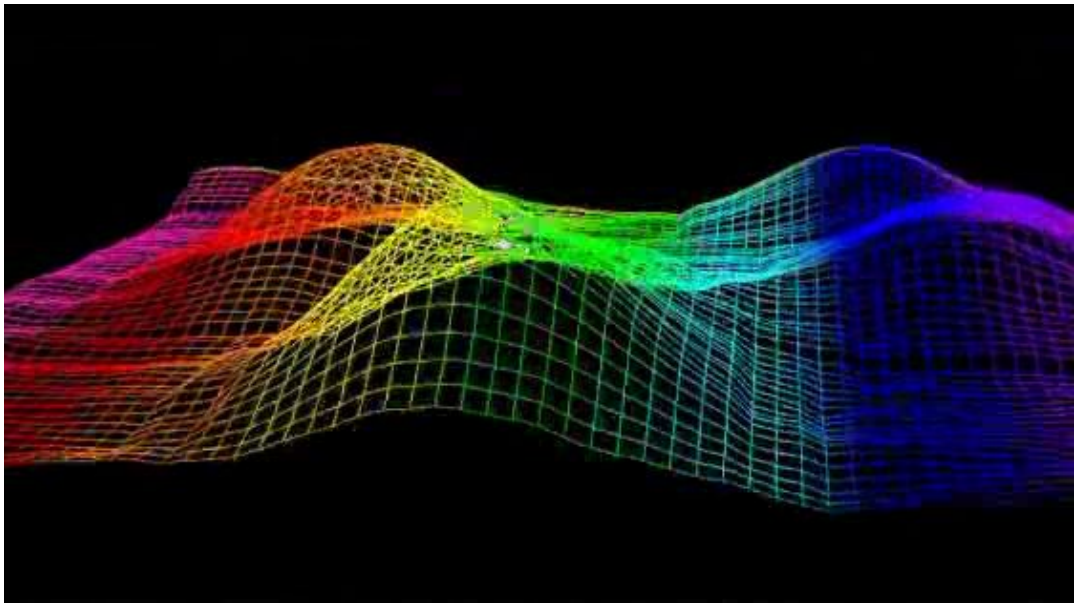
In order to place the data from each possible frequency at a coordinate on the terrain, it's planned that the center of the terrain will have the lowest frequency placed there, and that the other frequencies will be placed in concentric rings around the center. The largest ring will reach the edge of the terrain, and will represent the largest frequency of the FFT.

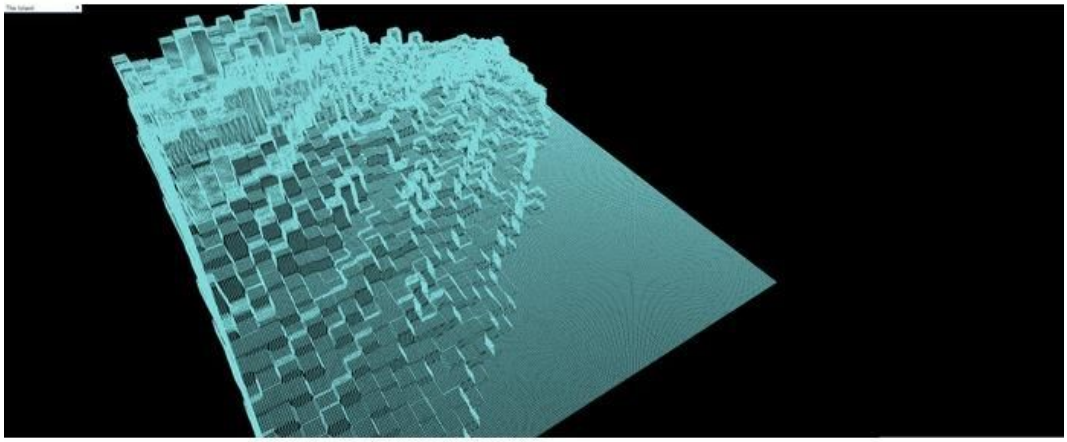
Additionally, I plan on being able to navigate around the terrain in a similar way to Assignment 2, i.e. rotate around the terrain, zoom in, etc. I'd also like to implement a pause button for the music that will stop the music but keep the terrain's heights and colors the same as when the music stopped.

Something that it'd like to do more research on soon would be to find ways to smooth out the FFT data to have it appear a little bit cleaner on the on the terrain. I'd also like to do research into using a framework like Three.js for this project. I'll have to do research on audio API's in JavaScript as well.

Sam McMahon
WebGL Final Project Report
CS 464
12/5/17

Screenshots of Similar Projects:





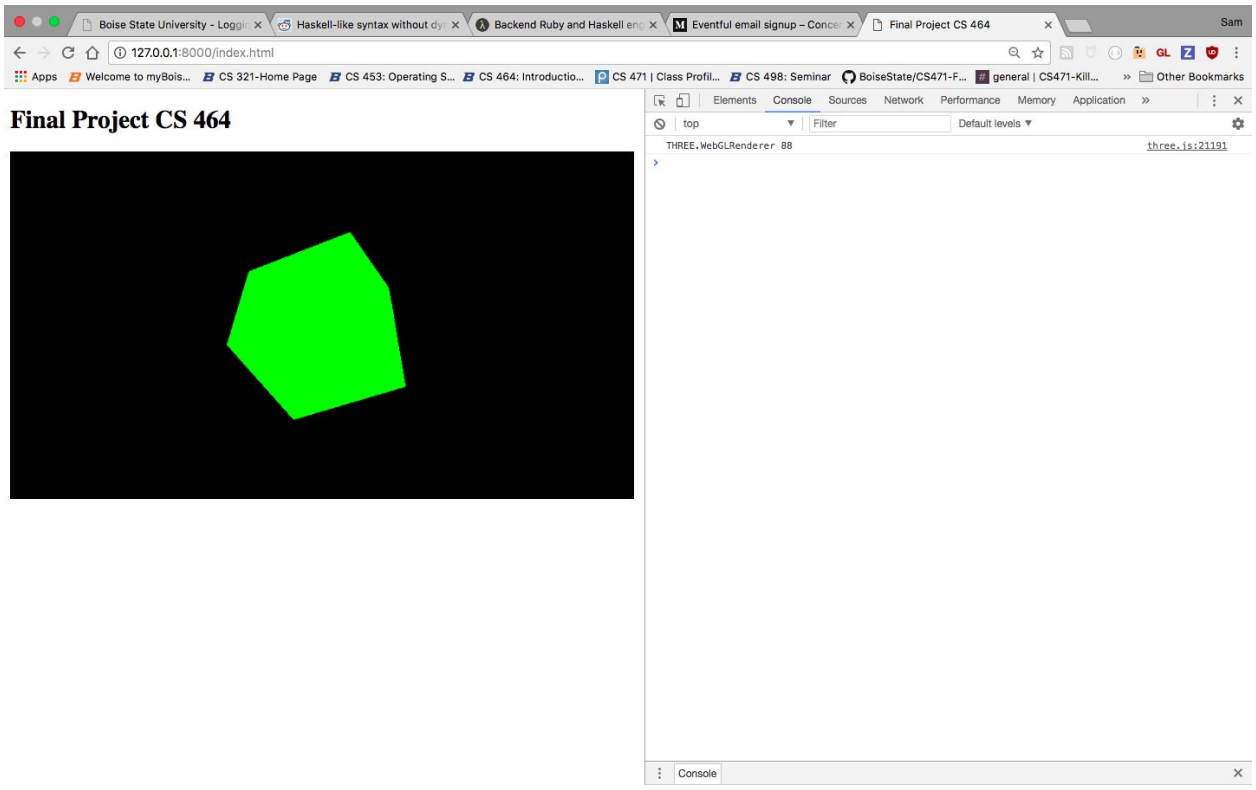
Sam McMahon
WebGL Final Project Report
CS 464
12/5/17

Status Report

In my project proposal, one of the things I mentioned that I'd like to do is explore using a framework like Three.js to do the project. I've been reading some documentation and tutorials on it

I was able to implement a (yet another) spinning cube demo using Three.js. It's pretty succinct, around 30 lines of JavaScript, vs. the 400 or so lines for the previous cube demos used in the class.

Another appeal of using Three.js is that it has an "AudioAnalyzer" object, which can load audio data and make calls to the brower's API for audio to get frequency data. I would like to explore using the AnalyzerNode interface to do other audio processing.



Sam McMahon
WebGL Final Project Report
CS 464
12/5/17

Final Report

What I implemented in the final version of the project was slightly different than what I envisioned in the beginning of the project.

The major difference is in how the visualizer mapped heights to the plane. The original idea was to have heights map of from the center of the plane, with the center representing the lowest frequency and the edge representing the highest frequency. This was changed to mapping the heights linearly across the surface of the plane, with low frequencies on the left and high frequencies on the right.

Using Three.js as a framework made implementing the project easier than I originally thought. Not only did it contain all the functions and objects I needed to render and the geometry to the screen in a concise way, but it also contained everything I needed to do the audio spectrum analysis. I'm really happy with how that portion of the project worked out.