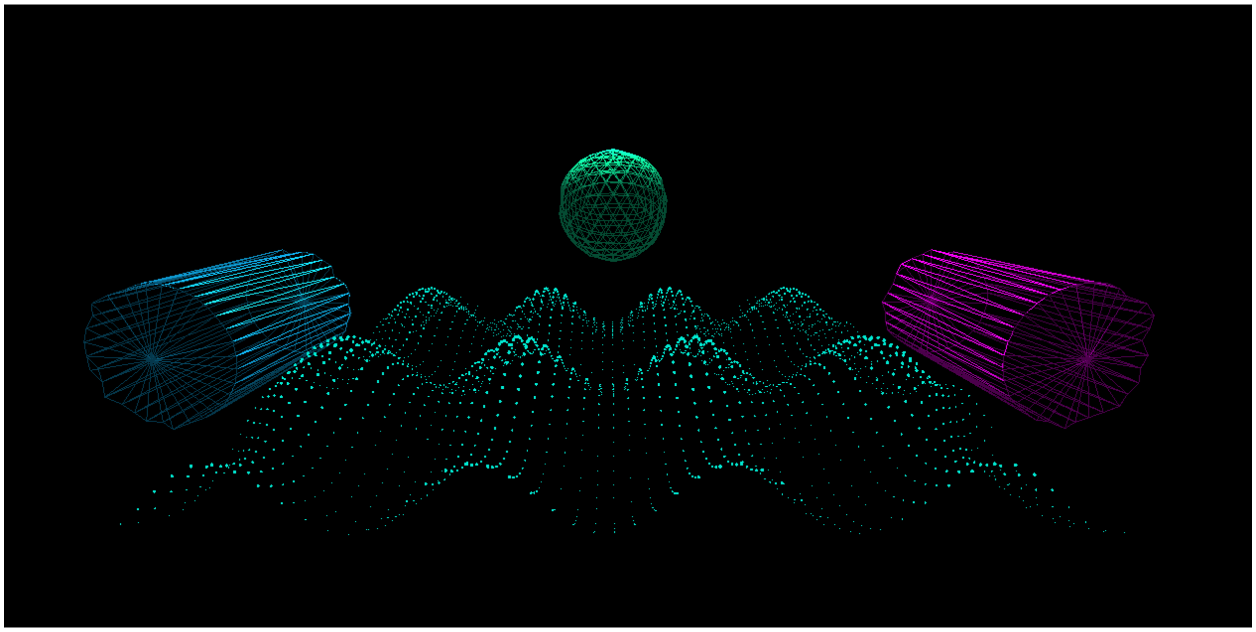
Program 5 - Audio Visualization



Names: Andy Choi, Sammy Siu

Files:
asg5.js
driver.html
Crystal_Castles_Vanished.mp3

Library files -
three.js
OrbitControls.js
dat.gui.min.js
simplex-noise.min.js

Requirements:

- A browser that supports WebGL (Google Chrome 9+, Firefox 4+, Opera 15+, Safari 5.1+, Internet Explorer 11 and Microsoft Edge)
- Audio file (.mp3 .mp4 or .wav)
- Audio output device

Description:
For our program, we made an audio visualizer using three.js and WebAudio API. The user interface consists of an audio player and a button to load an audio file. An audio file is provided. Camera pan and movement are done by the left and right mouse button while zooming is done by scrolling.

The scene starts with a system of particles, two cylinders, and a sphere. To start the visualization, click on the choose file button and select an audio file. By adjusting the audio volume, you can see how the models scale depending on the audio levels. You should see the models transform based on the audio.

Implementation:
WebAudio API is used to handle audio operations inside an audio context, which is the interface representing an audio-processing graph created from audio modules linked together, represented by an audio node. By creating the context, we are able to extract the attributes of the audio signal and use them to update the visuals of our models. To analyze the audio attributes in real time, we use the analyser node. The core concept of audio visualization is modifying the model's surface based on the vocals and beat signatures. In order to make our audio more profound on our models, we use procedural noise to add a more physical texture onto our models. To add noise, we used the Simplex Noise library.

Three.js is used for our 3D world creation. The models we used are particles, sphere, and cylinders. The scene is lit with a white spot light shining from the top and white ambient light at 0.3 intensity. Phong shading with wireframe is used for the cylinders and sphere while the particles are shaded with a uniform color of cyan. The particle positions scale off of the average treble decibel and the size scales off of the average bass decibel.

Reflection:
Using three.js is very similar to using webGL, while streamlining many of the tedious portions. It was much easier to create the scene using preexisting geometries. The lighting and camera were also much easier to set up.

A challenge that was encountered was the outdated nature of many of the references and sources we found, due to recent major changes to three.js. Another challenge was deforming the mesh through vertex manipulation in an aesthetically pleasing way.

Sources:

https://developer.mozilla.org/en-US/docs/Web/API/Web_Audio_API/Using_Web_Audio_API
https://medium.com/@mag_ops/music-visualiser-with-three-js-web-audio-api-b30175e7b5ba
https://threejs.org/docs/#manual/en/introduction/How-to-update-things
https://discourse.threejs.org/t/three-geometry-will-be-removed-from-core-with-r125/22401