**Project: P1**

\*\*\*old proposal paragraphs are highlighted

For our final project, we plan to design and implement an additive synthesizer. The minimum deliverable would allow the user to create unique sounds using synthesis techniques from one of a few select waveforms (sine, square, triangle or sawtooth) through a web application. The web application will have features and controls that are easy for a user to understand and use.

For our culminating project, we propose to make an additive synthesizer. The synthesizer would allow you to start from one of a few select waveforms (sine, square, triangle, or sawtooth), and then using synthesis techniques, create unique sounds. We plan to deliver this through a web application where all of these features and controls are appropriately represented as a UI to the user.

The techniques would include using LFOs (low frequency oscillators), ADSR (attack, decay, sustain, release) envelopes, and basic controls to manipulate properties of sound, like phase, pan, amplitude, resonance, distortion, etc. The frequency of the sound will be controlled through keyboard presses, corresponding to 1 octave of a piano (12 half steps, starting at A4), allowing you to play the sound you just created.

So far, we have implemented a master volume slider, pan, reverb, polyphony, and one of the low frequency oscillators (LFO) and one attack, decay, sustain, release (ADSR) envelope. The second row of the keyboard (Q through ]) produces 1 octave of a piano. (12 half steps, starting at A4). Furthermore, the user is not limited to only playing one note at a time, they can press and hold as many keys as they want to play a chord. The user also can adjust each of the properties we have programmed through basic UI that we have created so far.

In addition to these basic features of a synthesizer, we will add additional features if time permits. These features would include low, high, and band pass filters, common effects like delay and reverb, and the ability for the user to save their current settings and load old ones. Furthermore, the interface may be extended to be able to construct quantized notes, chords, and arpeggios.

We hope to continue implementing more on our synthesizer to allow users to manipulate sound. For our minimum deliverable we are planning on our synthesizer having another LFO and another ADSR envelope. We also want to have other controls to manipulate sound, such as phase, amplitude, resonance, distortion, etc. If we have time, allowing the user to save their current synthesizers settings and load old ones would make for a better real world application of our synthesizer.

We have created a github repository for this project. It can be found at: <https://github.com/leahdineen/audioProject>

The main resources we will leverage are included below. For low level digital signal processing, like performing Fourier transforms, and using oscillators, we will use the Web Audio API specification as an implementation interface. Also, the widely popular Synth Secrets series, written by synth expert Gordon Reid, contains many underlying details of additive synthesis, like using gates, envelopes, triggers, and more. Finally, Julius Smith’s Spectral Audio Signal Processing will be used for reference when doing the core implementation of our algorithms.

The Web Audio API included a function to generate the 4 basic waveforms, sine, square, sawtooth, and triangle. However, using this function prevented us from implementing some features that needed to modify the waveform, such as phase. To implement phase, I first needed to replace setting the waveforms with the API call with generating the waveforms with wave tables. The wave tables are 2 arrays that represent the real part (cosine coefficients) and the imaginary part (sine coefficients) of the wave. For each waveform I looked at the Fourier Series definition and set the kth element in the imaginary array to be the kth component in the sum. This is demonstrated in the following equations.

Sine

Square

Sawtooth

Triangle

Now implementing a phase shift of meant adjusting the real and imaginary arrays according to the following formulas.

**Resources:**

* Reid, Gordon. "Synth Secrets." Sound on Sound. July 2004. Web. 9 Feb. 2016. <http://www.soundonsound.com/sos/allsynthsecrets.htm>.
* Smith, Julius O., III. Spectral Audio Signal Processing. Center for Computer Research in Music and Acoustics. Stanford University. Web. 9 Feb. 2016. <https://ccrma.stanford.edu/~jos/sasp/>.
* "Web Audio API." Mozilla. Web. 9 Feb. 2016.

<https://developer.mozilla.org/en-US/docs/Web/API/Web\_Audio\_API>.