Music 257

LAB 4: SENSORS AS CONTROLLERS

Start thinking about your Final Projects! What controllers are you interested in? What concepts? Anyone you're interested in collaborating with?

I. GETTING STARTED WITH THE NEUROSKY AND UNITY

The Neurosky Mindwave Mobile is a EEG headset that you can use as a controller for Unity. It uses bluetooth, so make sure the computer your group uses has a bluetooth transmitter. If not, talk to Ethan and we can get you set up with a bluetooth dongle.

First, make sure your neurosky device has battery- turn it on and make sure that a little blue light is flashing next to the power switch. If this isn't the case, talk to Ethan about getting a new AAA battery.

Next, go to this page to walk through setting up your Unity project to work with the Neurosky.

Here's a quick rundown of what you'll need to do to set up:

1. Download the developer tools
2. Import their Unity Package into your project
3. Make sure bluetooth on your computer is turned on
4. Launch the TGC Installer in the ThinkGear Connector folder in the developer tools
5. Launch ThinkGear Connector
6. In your unity project, drag the NeuroSkyTGCController onto the game hierarchy.
7. Add the DisplayData.cs script as a component on your main camera.
8. On the Display Data script, change the size of the Signal Icons list to 5, and drag any three 2D textures onto each of the new fields there.
9. Hit play! The meditation and attention values should update on screen. To use these values to control the game, access the meditation1 and attention1 variables in the Display Data script.

II. IN A GROUP: USING THE MINDWAVE AS A CONTROLLER

To be shown off 5/3 in class

Create a Unity game that utilizes the Neurosky sensor to control the game state in some way. Remember that it will be nearly impossible for the player to actually control their current state in response to the game. Instead, you should design your game as a system that reacts to whatever state the player is in. E-mail Ethan a link to a 30 second pitch video and a video describing your game and how you used the controller. Make sure to include gameplay!

III. MATLAB: Due 5/5/16 at Midnight

Building filters. A filter is a system that you can feed any signal into and actively change the amplitude and phase of the sinusoids that build up that signal. There are four main types of filters. A *low-pass* filter is a filter that cuts off all of the high frequencies, while letting the low frequencies pass through. A *high-pass* filter lets high frequencies through. A *band-pass* filter lets a specific band of frequencies through, while cutting off all frequencies above and below that band. A *notch* filter notches out a specific band of frequencies, while letting through all frequencies above and below that band.

First, design a function that generates white noise:

```
[output] = generateNoise(duration, fs);
```

Generate noise will generate a list of random numbers between -1.0 and 1.0 that is `duration*fs` samples long. To do this, look up the `randn` function.

Plot the spectrum of the noise generated using your getSpectrum() function from Lab 3. Do this a couple of times, with a couple of different durations. What do the frequency components of white noise look like? Why does that make white noise a good way to test filters?

Next, look up the functions `butter` and `filter`. Note that you can use the *filter coefficients* (Two vectors- B and A) with filter to filter any audio signal.

Generate four spectrum plots and wav files of the following:

- White noise through a low-pass filter with a cutoff at 800Hz
- White noise through a high-pass filter with a cutoff at 1500Hz
- A song of your choice through a band-pass filter of your choice.
- A song of your choice through a notch filter of your choice.

Send Ethan your code, as well as these four plots and wav files.