Abby Schantz, Eliana Keinan, and Liz Harder

**CS35 Final Project Reflection**

Initially, we wanted to use what we have learned in CS35 and apply it to a hands-on, physical project. We decided to use two ‘toys’, a Sphero remote control ball and a Myo armband. While both toys are cool independently, we wanted to try to connect them, so as to control the Sphero ball using just our arm (the myo armband). We then wanted to turn this into a fun came which would allow the physical word to interact more directly with Python and the computer. Inspired by the image processing units in class, we decided to have pixels on an image change in relation to Sphero’s real life movements. We saw a number of versions of this and decided to pursue two paths. The first would be a drawing application in which as Sphero moved in the real world, he would draw onto whatever image was loaded into the file. The second was a game, in which there would be an image which appeared blank but actually contained hidden aliens under the surface. The user would have to move Sphero around in the real would to uncover the hidden aliens.

We quickly realized that just getting Sphero or Myo to interact with Python independently was going to be a challenge. Neither Sphero nor Myo have official SDKs to develop using Python. Thus, we had to use unofficial and quite janky SDKs to make it work. For Myo, this ended up not being too difficult, thanks to a well-developed, and more importantly, well documented, wrapper. Sphero was more complicated. The wrapper most people preferred only worked on Windows, which we discovered after a number of hours of debugging why it would not install. As it turns out, one of its dependencies is only available on Windows. Thus, we had to use a less popular wrapper, that is even more janky, to make it work.

Once we got Sphero and Myo interacting with Python independently, the task came to make them interact with each other. Because of the way the Sphero for Python was written, it ended up not being too challenging as it only required a listener. Thus, we were able to call commands within the Myo loops to make the Sphero ball move.

Independently from the robotic controls, we started developing our pixel games. For these, we used the tools we had learned in class, like openCV to modify pixels within an image. Harder than the actual code writing, was figuring out in our minds what we wanted to happen. For the alien game, for example, did we want to be replacing the pixel, or changing a pixel to be transparent? We ended up replacing pixels as that aligned more closely to the work we had previously done using the program.

Finally, we had to combine all three steps. Due to time constraints, this was not quite as fleshed out as we would have hoped but we did get it working! At first, we struggled to get all three programs talking to each other. When one would run, it would cause the other to shut down or at least stop listening to commands. Professor Dodd’s helped us figure out where to place the code within the Myo and Sphero file to get it to interact correctly.

At this point, we have accomplished all of our primary goals. Sphero and Myo both interact with Python. You can control Sphero using your arm and you can see Sphero’s movements by drawing on the screen. Alternatively, you can see Sphero ‘erasing’ the pixels to discover hidden elements in a picture as you move him in the real world.

If we had more time, we would improve these two games considerably. At the moment, it does not draw clean lines. It moves at an arbitrary distance. Additionally, a game board is not defined so it is easy to roll Sphero too far in the real world such that his pixels will have no effect on the image. There are also many elements to the games we could add, such as timing a user’s play, having pixels reappear after certain amount of time, or giving the user the option to turn draw on and off (lift the pen) so he or she can move Sphero in the real world so create real images, instead of just scribbles. Additionally, two switch between games right now, the user has to comment and uncomment the code of the other game. We should switch this to be more user intuitive.

Overall, we are thrilled that we got all the components working and had an amazing semester exploring Python and all of its fun and exciting libraries.