Daniel Thirman

Spring Quarter Summary

For this past quarter I have been investigating different basic control modalities for the third arm. The two main methods I looked into were head gesture control and arm gesture control. Each one had different benefits and drawbacks and were better for different scenarios. Since I am currently unsure which one is better to the average person I have designed a study to test how well users react to the different methods and which one would e preferential for the third arm.

To implement head gesture control I used an Adafruit Flora which I mounted on a beanie. The Flora miniature microcontroller designed for working with wearable technology that runs on the Arduino software library. In addition, I used a 9 degree of freedom sensor which connects to the Flora over I2C and reads the current accelerometer, angular rate, and magnetometer data. The first gesture that I wanted to recognize was a simple yes/no head gesture. My first attempt in doing was to use a support vector machine, or an svm. An svm is a simple classifier that separates and classifies data points by maximizing the distance between the decision boundary and the points closest to that decision boundary. The svm had three output classes yes, no, and no gesture, and its input vector was a vector of output of the past five readings from the Flora. To train the svm I recorded data of myself repeatedly moving shaking my head yes, shaking my head no, and performing no gesture but moving my head around as normal. I experimented with which parts of the data I should use and found that the best results came from only using the angular rate sensor. Accelerometer data tended to make results worse as it does a poor job recognizing a no gesture, since the device is merely turning but not in a way that is distinguishable by the accelerometer. The magnetometer was also not included as the direction that one is facing should not impact whether or not they are performing a head gesture.

Once I had the svm implemented, it had reasonably strong results. It did a good job differentiating yes from no and would always recognize when a gesture was performed, however it was still overly sensitive and tended to record gestures when there were none and instead there was just regular head movement. I reduced some of this noise by setting speed thresholds that must be reached to obtain a head gesture, however there was still some noise. Since the results were reasonably good I decided to move on to the next part, a simple way of moving a virtual box simulated with Unity around a screen while making use of head gestures.

The first method of moving the box that I implemented was that where the box would move to the position on the screen where the user was looking. The location where the user was looking was determined by reading the magnetometer data from the Flora and then converting that data into a Euler angle. In addition to moving the box by looking the user could lock or unlock the box from moving by performing the yes gesture to unlock it and the no gesture to lock it in place.

The second method was fairly similar to the first in that it still had the same movement locking and unlocking controls and still read the direction the user was looking. However, instead of the direction setting the position of the box it set its velocity, so the farther the user looked in one direction the faster the box would move in that direction. I implemented this method in one and two dimensions, and found that it worked relatively well, however two dimensional movement was significantly more noisy than either of the two dimensions by themselves. I also tried running the simulation with and without gravity to see whether how natural it felt. I found that these results were relatively good and the user was able to have the box move in the direction that they wanted it to with little trouble. One of the biggest modes of failure was that moving one’s head around to move the box would frequently register as a yes or no head gesture and lock the box from moving, so I decided to reinvestigate head gestures to see if I could find a more stable method.

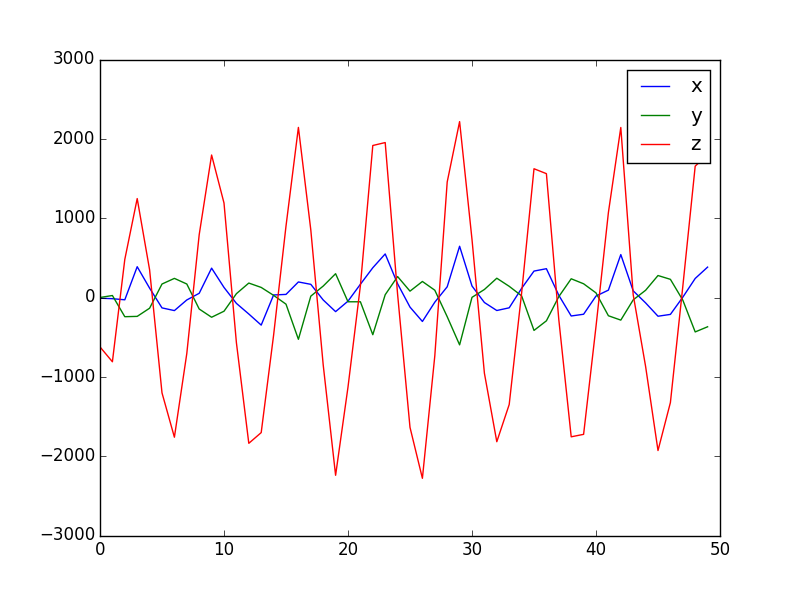
At this point I recorded the angular rate data that is recorded by when the user performs a yes or no gesture, I have included the graphs below.

Figure : Yes Gesture

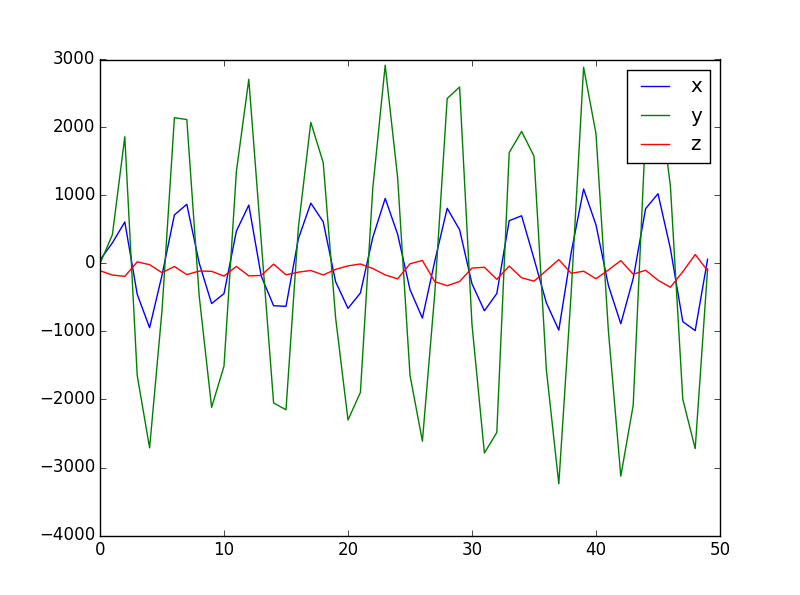


Figure : No gesture

As can be seen in the graphs, each gesture is relatively similar. For each yes or no the head reaches a similar maximum and minimum angular rate on the axis in which the gesture is performed. So, I implemented a new method which was simpler than training a full svm. In the new method the user would first indicate yes several times and then no several times to train the classifier to learn how large their yes’s and no’s are. After the initial training the user would move their head around as they normally do, then if during any three second window they reach 50% of the maximum and minimum recorded during the training period then it would be registered as a yes or no. The results produced by this method were incredibly good. It recognized nearly every gesture performed and there were almost no false positives in gesture reading as there had been before. So, at this point I decided to move on to arm gesture control, so I could compare it to head gesture control.

To implement the arm gesture control I used the Myo armband. The Myo armband is a sensor that reads EMG data measured from the arm where it is mounted. In addition to returning the EMG data it also returned the Euler angles of the arm, and whether or not the user had performed a hand gesture. The hand gestures it recognized were spread fingers apart, tap fingers together, swipe left, swipe right, and create a fist. While it would have been possible to use two of the gestures, such as swipe left and swipe right as a method of determining yes and no, this felt unnatural and confusing, since the user would not be used to performing that gesture to indicate yes or no and would most likely get confused. So, I decided to determine the users input by looking for a thumbs up for yes, or a thumbs down for no. While the Myo did not have a built in way to determine thumbs up vs thumbs down, it was possible as long as the user was somewhat tricked. Since the Myo was already looking for a when the user created a fist, which is created whenever a thumbs up or thumbs down is performed, it was possible to detect by checking the roll of the arm when a fist was created. The user would be told to raise their thumb while performing this gesture, however that would actually not be necessary and they would able to simply tilt their closed fist and create the same results. This implementation proved to be decent, for some users it worked very well with high accuracy, however others had trouble using the Myo which would fail to record their gestures unless they squeezed their fists particularly tightly. Since I did not know which of the two different control modalities would prove to be the most natural or useful for the average user I decided to design a study to test it.

In the first iteration of the study I designed the user would be put through a Stroop effect like test, in which they see a color word written in a color and they would have to indicate whether or not the two colors matched. However, I decided to change this to simply telling the user which gesture to perform as that is a simpler test which tests the user’s ability to input gestures more directly. Some users of the test would only have the option to use head gestures, some would only be able to use arm gestures, while a third group would have the option of using both methods and would use the one of their choice. I will measure the reaction times and accuracy rates for each group or users and use that to determine what the best method for input is. At this point the study is fully implemented and I am ready to run the tests on users and will do so over the summer. Once I have drawn the conclusions from the study I will look into more complicated gestures and actions and move to more complicated simulations to better simulate controlling the third arm.