ME 410 - Week 4 Summary

Logan Boswell

Summary

This week's objectives were to assemble our drone and a pitch angle testing setup, implement pitch angle proportional, derivative, and PD control on the drone, and test our drone with different filter bandwidths.

After assembling the drone, we used the provided fishing line and c-clamps to set up a test rig where the drone could be held in place and freely rotate about the pitch axis. With this testing setup, it is easier to test our pitch angle controllers, since the drone rotation is isolated to only one axis.

Before we started implementing controllers, we added a line to our code to actually send the calculated motor speed commands to the motors. After this, we tested a proportional controller (gain of 10) by moving the drone by hand and seeing if the thrust-induced torque was applied in the proper direction to correct the drone's orientation. The plot shown below was generated to show the effect of the proportional controller.

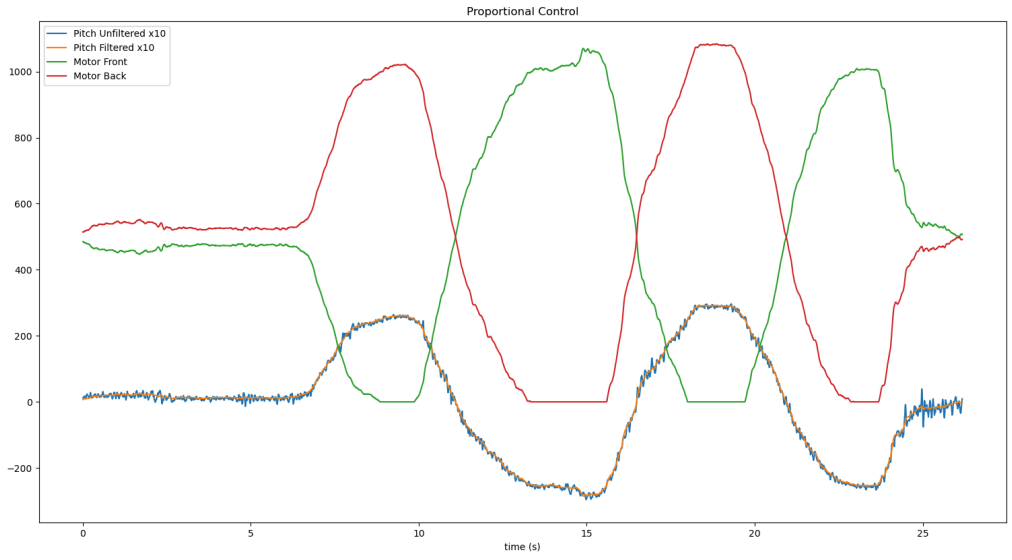

Figure 1: Proportional Controller Test

Next, we moved on to test a derivative controller (gain of 1) by rotating the drone and seeing if the thrust-induced torque opposes the motion. The plot below shows the effect of the derivative controller.

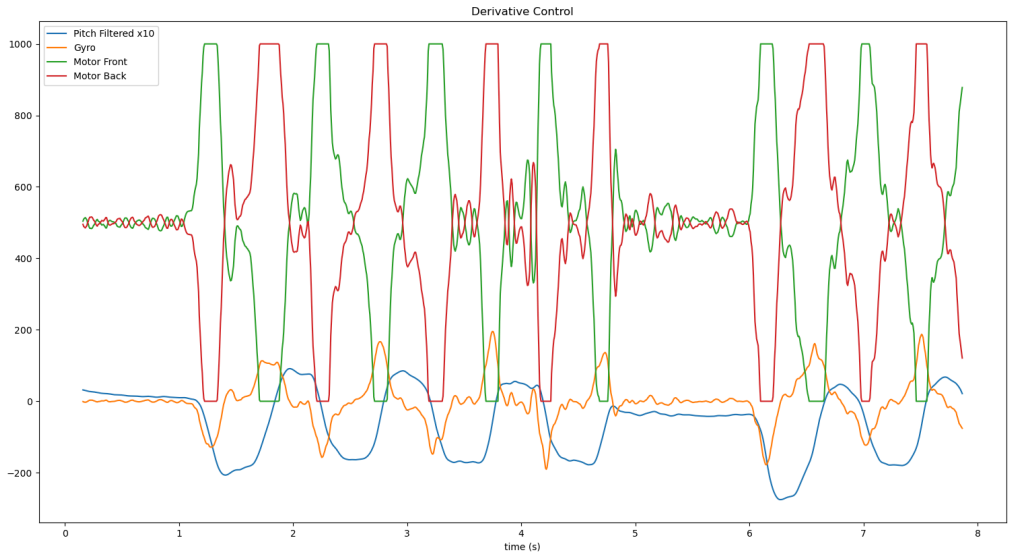

Figure 2: Derivative Control Testing

After testing both proportional and derivative control individually, we then tested a PD controller with a proportional gain of 15 and a derivative gain of 4. In this test, the goal was for the drone to hold itself at a set position and smoothly track a desired angle from the joystick. In our case, the battery on our drone was off-balance, resulting in our drone being heavier on one side compared to the other. Since we do not have any integral control, there is steady-state error due to this imbalance, but our drone is still able to move accordingly based on the joystick input. I think our performance can be improved by further tuning, but we were still able to track an angle. The plot below shows results from testing our PD controller.

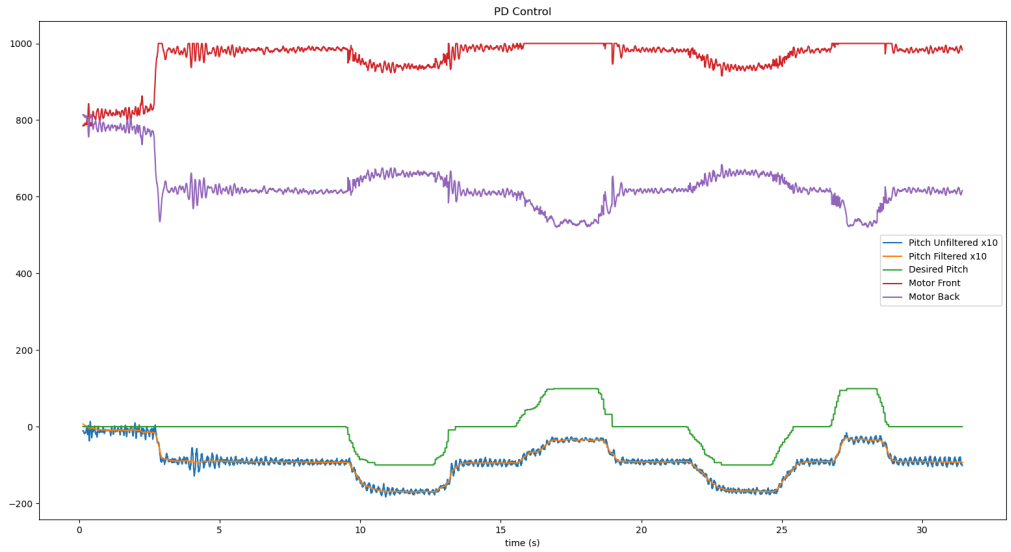

Figure 3: PD Controller Testing

Lastly, to show the effects of changing filter bandwidth. Three tests were performed, one with the normal filter setting (0xA9), one with 2-fold oversampling (0x99), and one with 4-fold oversampling (0x89). 4-fold oversampling yielded the best results and the plots below show the noise for each of these settings.

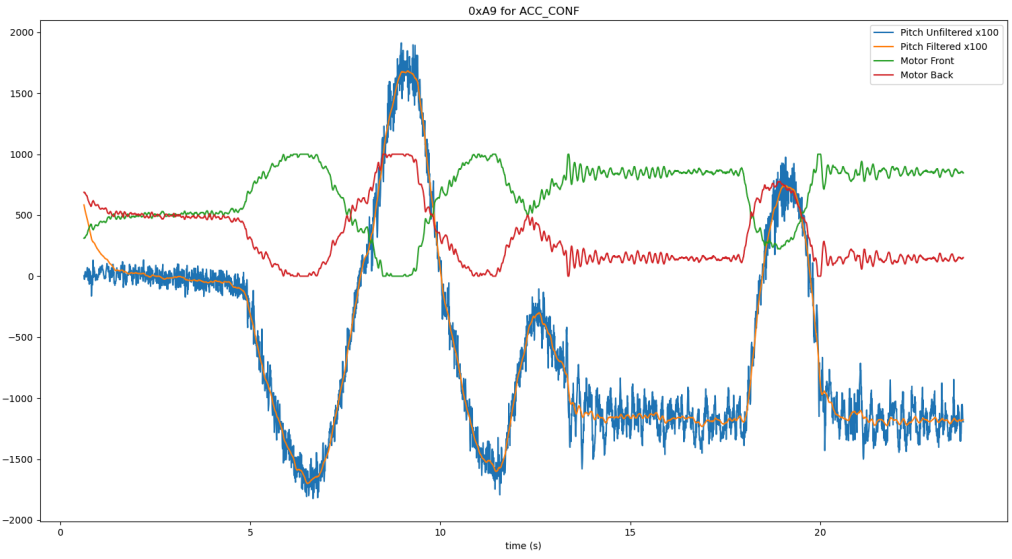

Figure 4: Normal Filter Setting

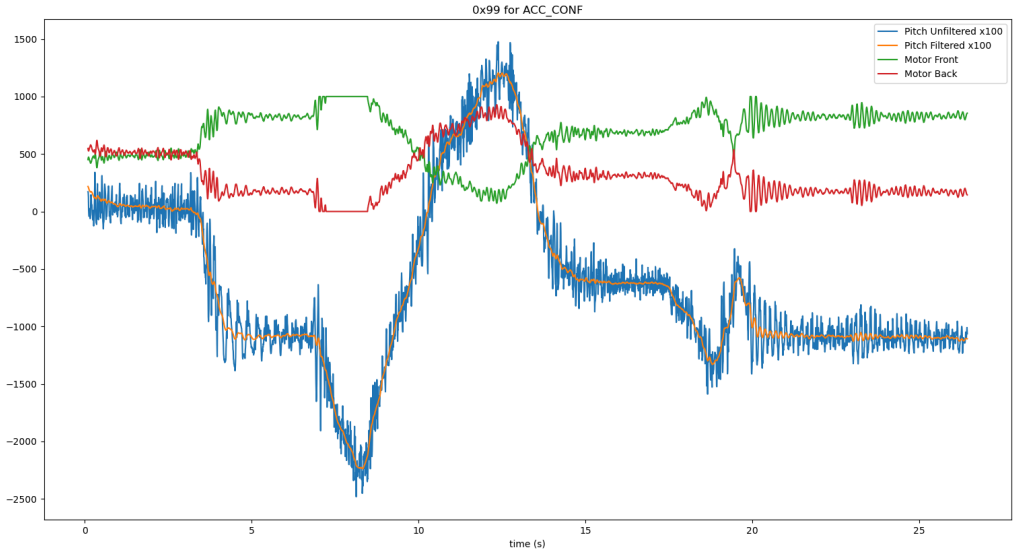

Figure 5: 2-fold Oversampling Setting

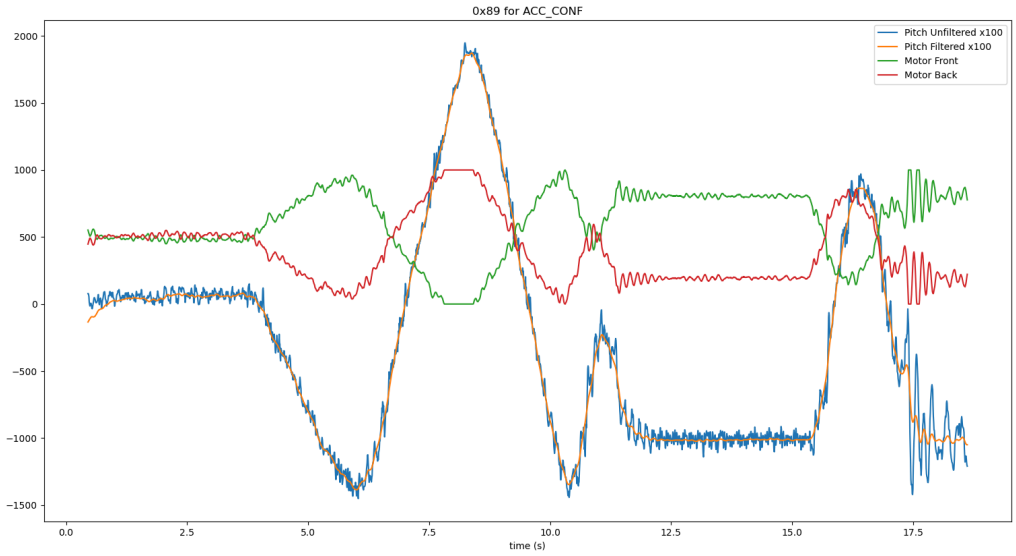

Figure 6: 4-fold Oversampling Setting

Assessment - What Went Well

This week, I feel like tuning the controllers and plotting went pretty quickly and smoothly. For the controllers, we did not have to spend much time tuning before we got an acceptable result. For the plotting, we used a USB drive to quickly move the csv files used for plotting from one computer to another without having one person disconnect from the drone's wifi and commit to github.

Assessment - What Did Not Go Well

This week, we really got held up on setting up the test rig. We had a working setup, but our wires were kind of long, so we took our setup apart and tried to make the wires shorter. This probably took us maybe 30-45 minutes, so we wasted a lot of our in-class time.

Assessment - Adjustments for Next Class

Overall, I think the main adjustment we need to make is not wasting time trying to make the testing setup perfect. We currently still have our c-clamps tied with our wires, so hopefully we do not take as long setting the rig up.

Team Member Effort

Me - 49% (Worked on plotting code on my laptop after collect data as csv file; helped Ben with editing our .cpp file; helped with testing setup)

Ben - 51% (We worked on his laptop to edit our .cpp file; saved our data to csv file; helped with testing setup)