Matthew Harter
Travis Stanger

ENGR 454 – Project Milestone 7

Due: 5/10/2021

Process:

In hand placing the poles, we used the place() function in Octave. We did this in a Jupyter Lab IPython Notebook with the Octave kernel. We set tried setting different poles to find the gain values that worked the best for our system. With poles = [-10 -5] (first value for position and second for angular velocity), we achieved the best gain with G = [0.1720 0.1129] .

From our thrust modeling we found that the hover copter balances when the motor is producing about 40 grams of thrust which corresponds to a PWM value of about 80 which we set as our large signal PWM. Our small signal PWM is given by $-G \cdot X \cdot 1023$ where G is our 1x2 gain matrix and is our 2x1 state variable matrix. The purpose of the 1023 multiplication is to scale the gain values over the possible range of PWM values. The small signal is added to the large signal PWM after each read of the rotary angle encoder. The period between reads is calculated by storing the time, using the millis() function, at which data points are recorded/calculated into lastTime and then calculating deltaTime = millis()-lastTime where delta time is the period. Our initial code contained no added delays and ran the data recording and calculations as fast as possible. With no delay the system failed to settle and would instead overcompensate resulting in larger and large ripples (see video). By adding a delay of 1ms the system responded much better overall and settled much quicker. We believe that the adding this delay allows the system to have more time to react and settle on a value before the next is taken.

Results:

With the poles set as mentioned above and the calculated gain values for the position and angular velocity, our system settles very quickly—within 2.5 seconds (see video).