Quadcopter Simulation/Real-World Implementation

Ryan Rafati and Karl Simon
MAE 6245
Professor Nikhil Nigam
George Washington University
April, 30th 2024

Code Repository:
https://github.com/KarlSim24/mae6245_quadcopter.git

Quadcopter Dynamics and Simulation:

Karl: We first spent time researching quadcopter equations of motion, dynamics, and simulation code to learn more about how we wanted to approach the project. We first looked at [1] "Quadcopter Dynamics, Simulation, and Control." This paper provided us an overview of how a complete quadcopter model and controller is laid out, but went a bit too much into the details of real-world characteristics of a quadcopter (such as EMFs and power/battery constraints). We also referenced [2] eluu11_public.pdf which was similar, but had a confusing convention for linear and angular positions. We mainly used the second resource to confirm whether portions of the dynamics derivations we were doing were correct or not.

Next, we followed a course from UIC https://pab47.github.io/robotics.html about how to derive the dynamics of the quadcopter. We ensured the conventions used matched the conventions from our own course, as well as the textbook [1] mentioned above. The derivations we were able to compute are in the following notebook images:

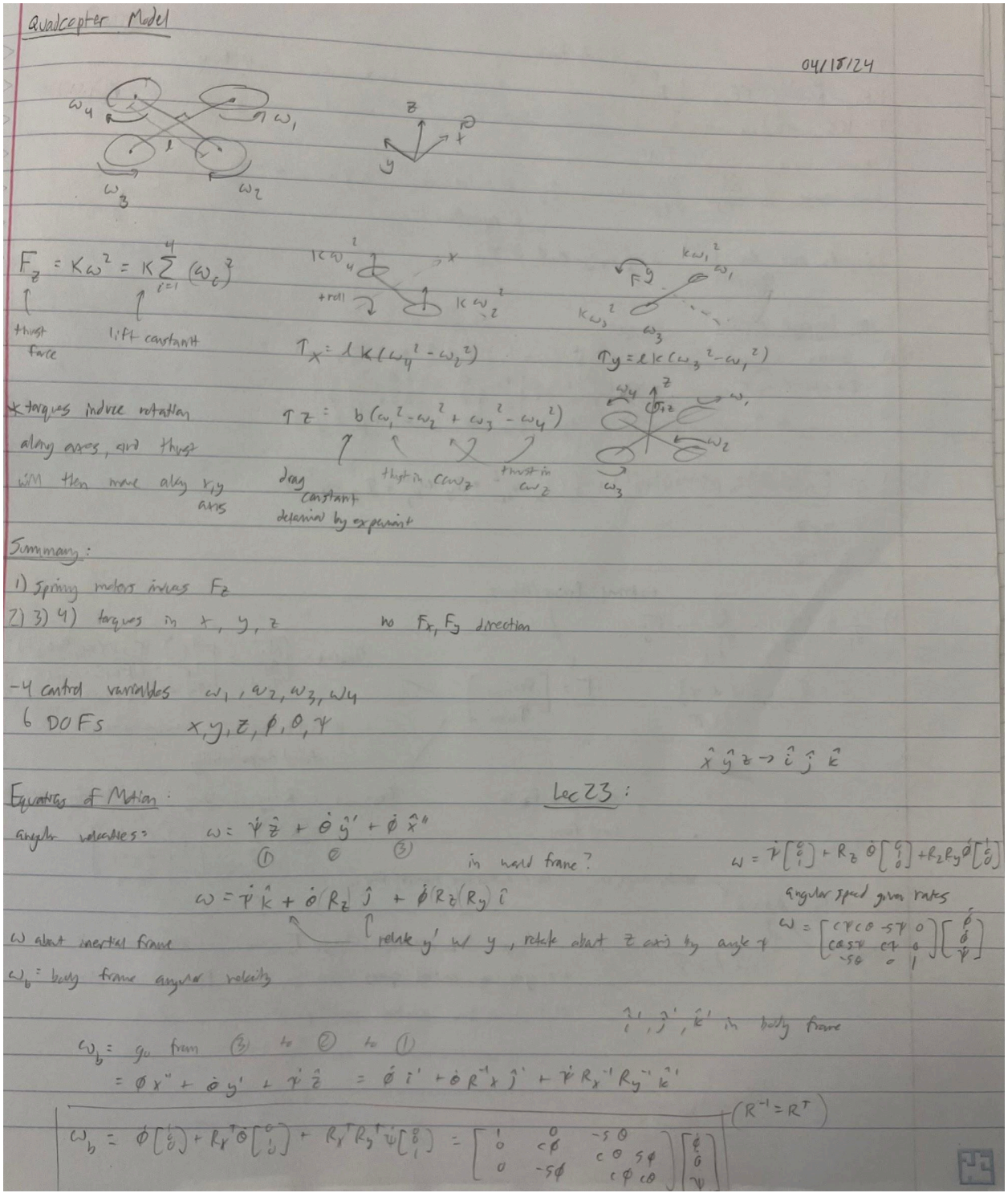

Figure 1: Quadcopter model

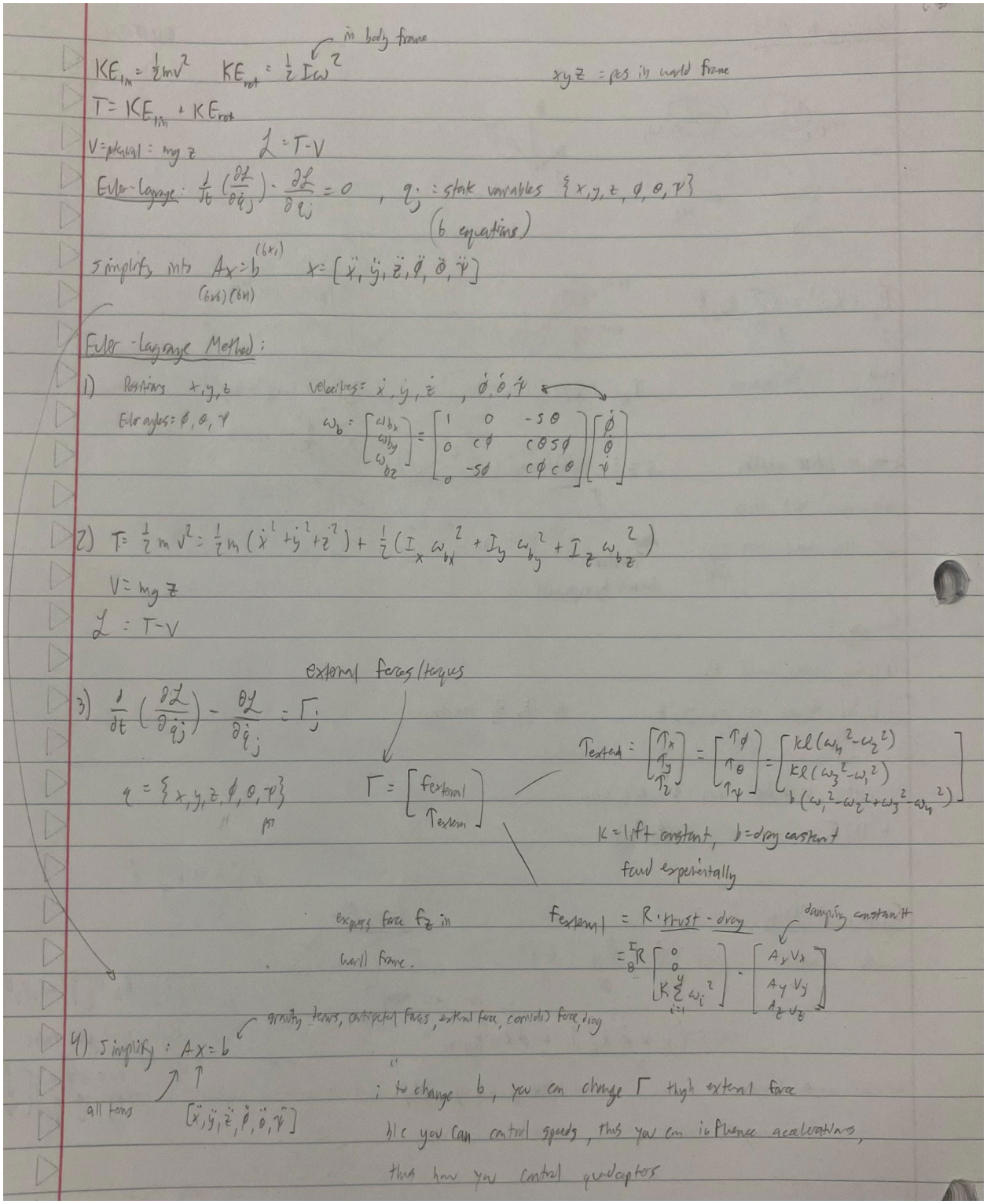

Figure 2: Quadcopter Dynamics and EOMs

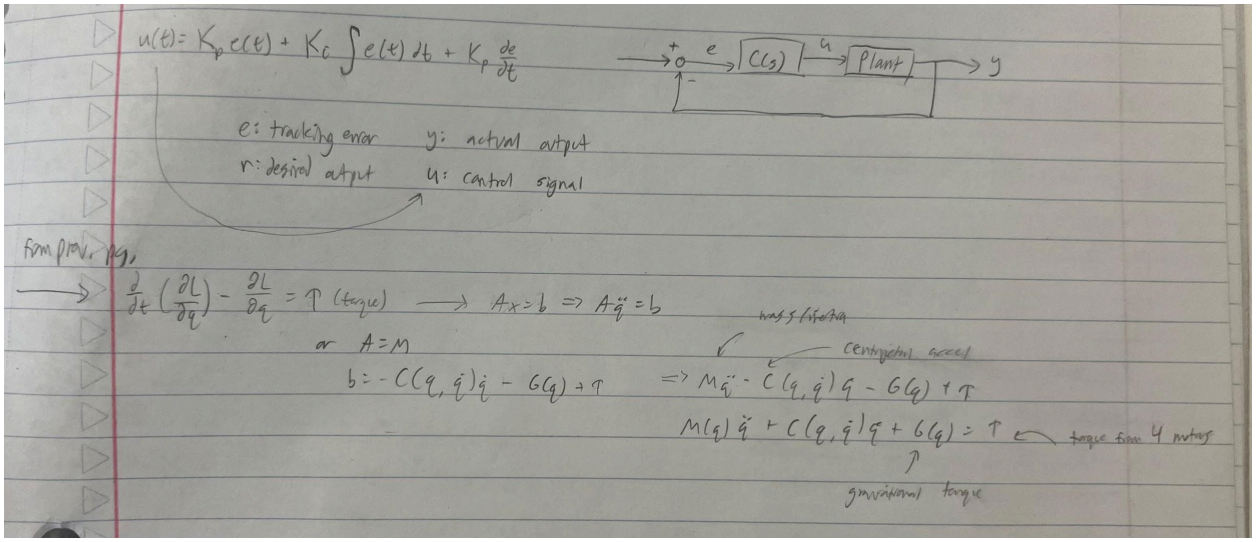

Figure 3: Started, but not finished controls setup

Next, we used matlab, following the linked course resources, to implement the simple quadcopter model in a simulation. *Note that we borrowed parts of our code from the UIC course.* The two files are equations_quadcopter.m and sim_quadcopter.m. To run the simple code, run `>> sim_quadcopter.m`. The code will show the quadcopter rotate along the body x-axis, however since there is no controller, it will continue to roll around this axis indefinitely:

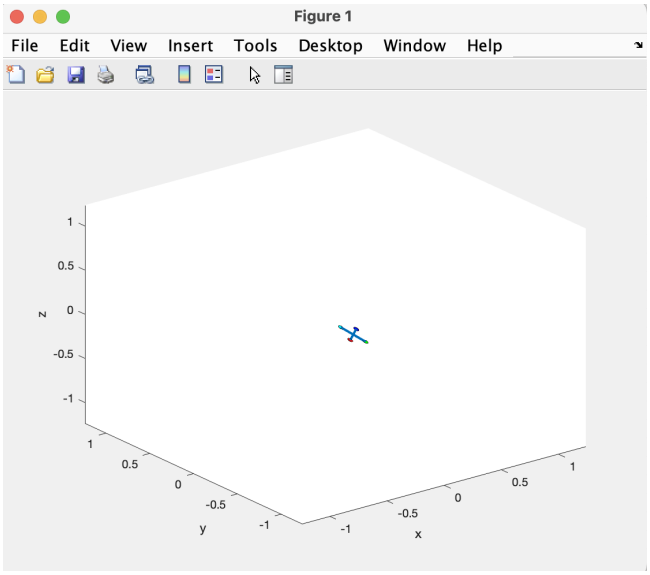

Figure 4: simulation window

I had tried to implement a PID controller for simple attitude control of the quadcopter in the simulation, however I realized that it was more challenging than expected. For simple 1D cases, PID controllers are understandable for me (such as the one I followed here). However, the quadcopter was highly-coupled, which made me realize that implementing one was beyond my current skill level. That said, this is a project I will continue to work/return to on once I have a more comfortable background in controls.

MAE 4182 - Electro-Mechanical Controls (Professor Lee)

Ryan: In the Fall of 2023, I took a Control System class with Professor Lee. This class introduced me to the control library in Python among other functions to help me learn different characteristics of control systems, like Stability, Block diagrams, Root Locus, and many other concepts. For this project, I went back and reviewed my class notes and the different control methods we learned. I was particularly interested in understanding the PID controller we covered in class and how gain values can change the outcome. This approach utilizes some of the concepts learned in this class while integrating the simulation and PID controller into the simulation shown in "Quadcopter Dynamics, Simulation, and Control." This document also furthers my knowledge of PID controlling mechanisms and equations of Motion regarding the quadcopter.

Our process is outlined in the following block diagram of a PID controller.

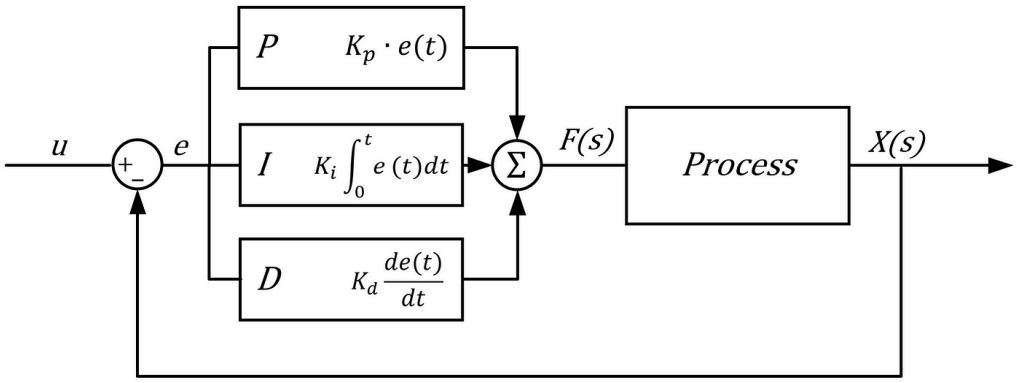

Figure 5: Block Diagram - PID controller
(Source:https://medium.com/@jadenbh12/what-i-learned-from-building-my-second-quadcopter-an-in-depth-breakdown-of-pid-controllers-3860964dc75)

In our Bode Plot Python notebook, the 'pid_controller' function has its numnertors labeled as the gain values for the proportional, integral, and derivative blocks in the diagram. The transfer function of these blocks is given by diving the gain values, by the 'plant' which is a simple array. Initial mechanical parameters of the quadcopter are defined and are used in a transfer function. The 'Quad' variable represent the plant of our system and the equation has been adapted from my previous course, with changes to some of the parameters. This accounts for the second order system where the variables can influence the dynamics. This plant does not include a controller, which then implemented in the next step.

Selecting gain values was a challenge for this plot, as the understanding of how each block inlfuecens the output function is complex. Gain values were selected based on previous notes and lectures given in MAE 4182. Utilizing the control library a bode plot was created to display frequency response of the controller. Analyzing the plots it is determined that there are issues with the stability of the system as when the phase decreases there is a lag that could be

improved with tuning. Also shown in the notebook is a data function with multiple characterics of the plot and controller. The system is unstable since there are negative poles, and there are also other errors as many of the numbers are extremely high.

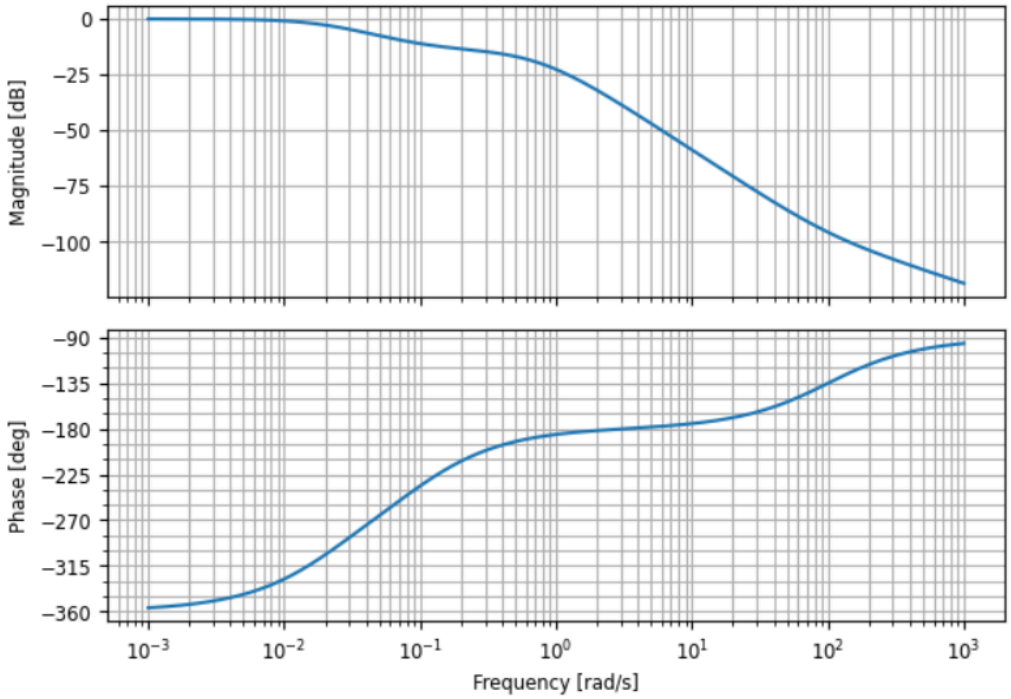

Figure 6: Bode Plot

The “Flight_Path_Simlation” utilizes a combination of MAE 4182 code, and code from “Quadcopter Dynamics, Simulation, and Control.” It was assisted from watching multiple different tutorials and other research online. This simulation incorporates the same PID controller developed in the previous example, however includes a ‘loop’ for time intervals. The controller also integrate error that could be posed in the controller. Playing around with different gain values can show how the flight path be changed.

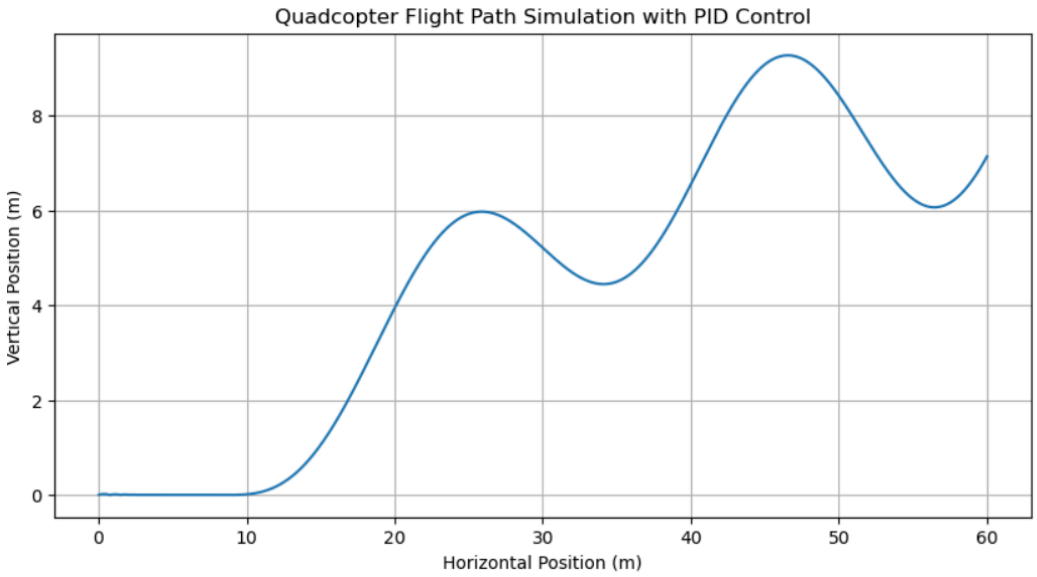

Figure 7: Flight Path

Real-World Application:

As mentioned previously, Karl and I were in the same group for our final senior design project (which won the best capstone project award). This project entailed designing an Unmanned Aerial Vehicle capable of autonomously detecting arco markers throughout a series of challenges outlined by our sponsor Raytheon Technologies. The project involved many hours of test flights and simulation efforts throughout the course of each semester. We competed in a competition from **04/25 - 04/28** at Virginia Tech's with other schools across the nation. Our system dived deep into understanding the trajectory planning and dynamics of the hexacopter, and also techniques on how to integrate hardware and software effectively. Having taken this course has greatly helped us understand how coordinate frame transformations and model dynamics are used by the flight controller to move the UAV around the physical world.

Figure 7: Autonomous Hexacopter in flight

Figure 8: Computer Science UAV team configuring on board Jetson computer

A successful flight and target (with terribly tuned controller gains):
https://youtu.be/dCvVSPeg8U4