CEG5303 MiniLab Report

Group 45

WEI TAO (A0296728J)

XIONG RUIYI (A0297876Y)

YANG PEILIN(A0297046W)

March 17, 2025

Contents

1 Introduction

In this experimental assignment, we used Matlab and Simulink to model the quadcopter system according to a simplified dynamic model. We implemented a PID-based control method and adjusted the parameters to control the quadcopter to follow a given trajectory. We changed the trajectory and environment of the operation and adjusted the corresponding parameters to better master the PID control method.

2 Group: Basic Realization

Following the instructions in the guidance document, we successfully built a drone simulator using the PID control method that can reasonably follow a predefined simple trajectory.

2.1 Analysis of dynamic principles

This Simulink structure diagram is an implementation of a quadcopter control system, which is mainly used to control the drone to fly according to a reference trajectory. The entire system consists of multiple modules, including position control, attitude control, altitude control, yaw control, and a quadcopter dynamics model.

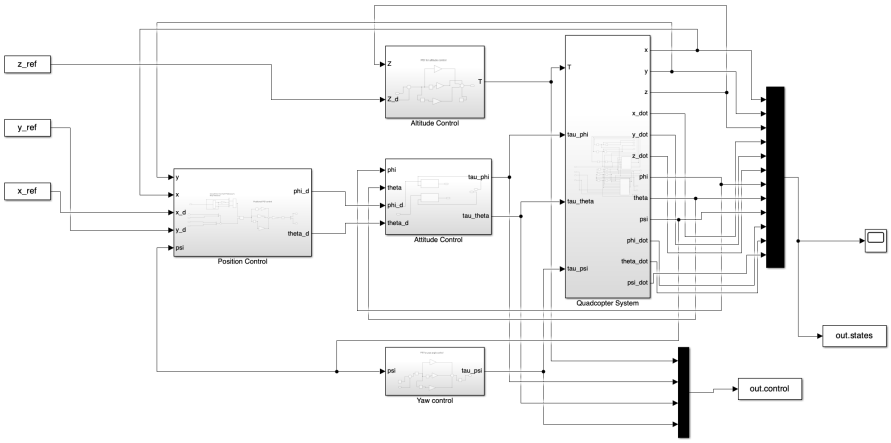


Figure 1: Quadcopter control system Simulink architecture.

x_ref, y_ref, z_ref are three input variables representing the target position (reference trajectory) of the quadcopter in 3D space, which indicates the desired position of the UAV in the x, y, z directions.

2.2 Position Control Module

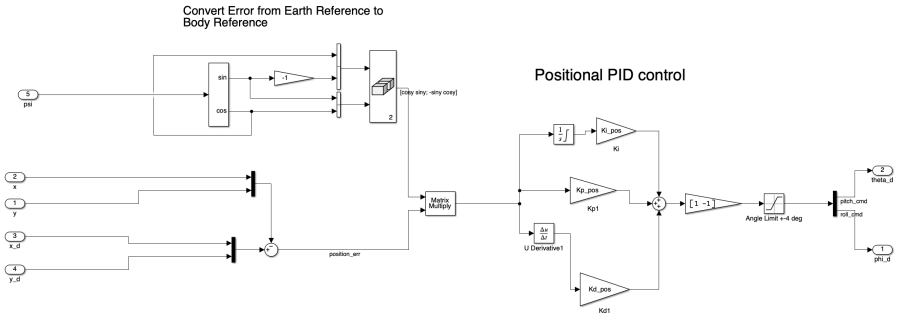


Figure 2: Position Control Architecture

This figure illustrates the implementation of the Position Control module, including the error transformation from the Earth Reference frame to the Body Reference frame, as well as attitude control based on PID control.

2.3 Altitude Control Module

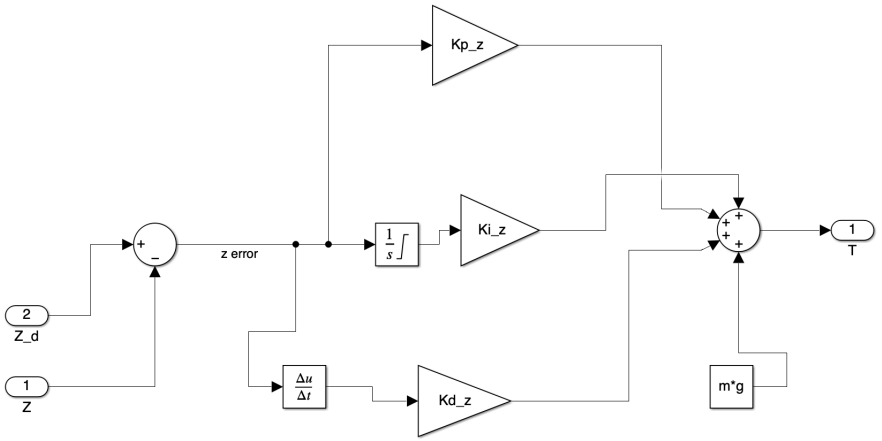


Figure 3: Altitude Control Module

This section of the PID controller computes the altitude error and applies Proportional (P), Integral (I), and Derivative (D) control strategies to adjust the UAV's thrust T, ensuring that the UAV maintains or reaches the target altitude Z_d.

Additionally, a gravity compensation term $(m \cdot g)$ is introduced to account for gravitational effects, making the thrust calculation more accurate and stabilizing the UAV during flight.

2.4 Attitude Control Module

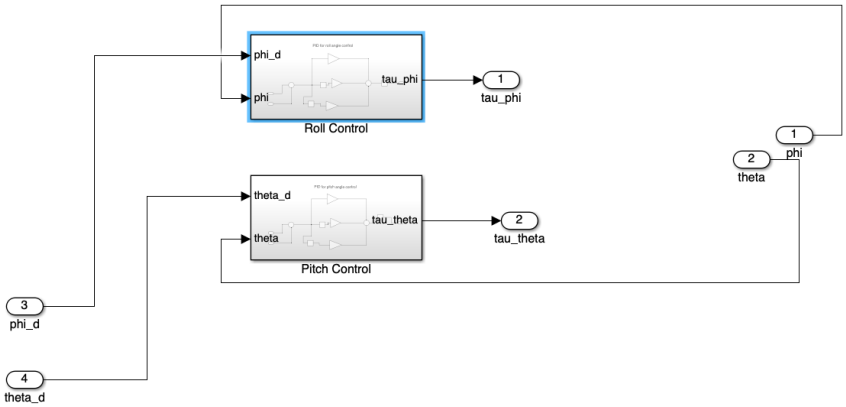


Figure 4: Positional PID Control

This module utilizes a PID controller to compute attitude errors and adjust control torques, ensuring that the UAV's actual attitude (ϕ, θ) gradually approaches the target attitude (ϕ_d, θ_d).

The **Roll Control** and **Pitch Control** components are responsible for separately adjusting the UAV's roll and pitch angles to ensure stable and precise flight.

2.5 Yaw Control Module

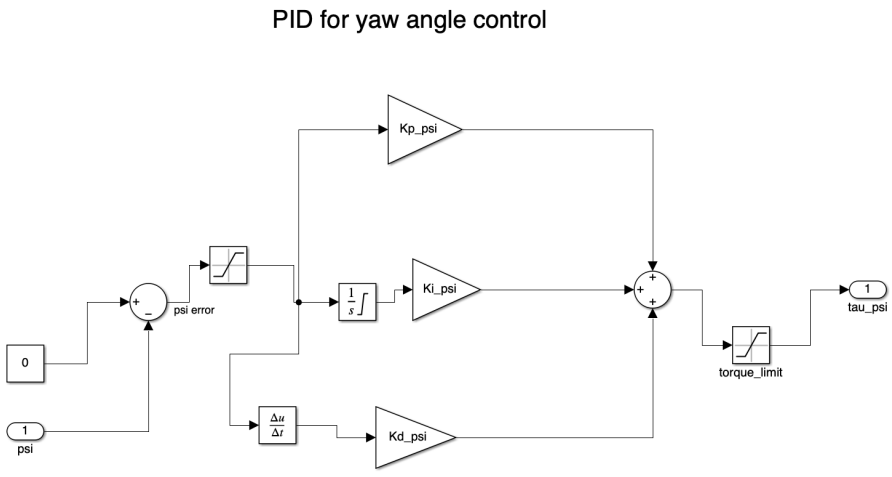


Figure 5: Yaw Control Module

This control system is designed to regulate and stabilize the UAV's yaw angle (ψ). By applying a PID control algorithm, it precisely adjusts the control torque, ensuring

that the UAV aligns accurately with the desired direction, preventing rotational drift, and improving flight stability.

2.6 Quadcopter System Module

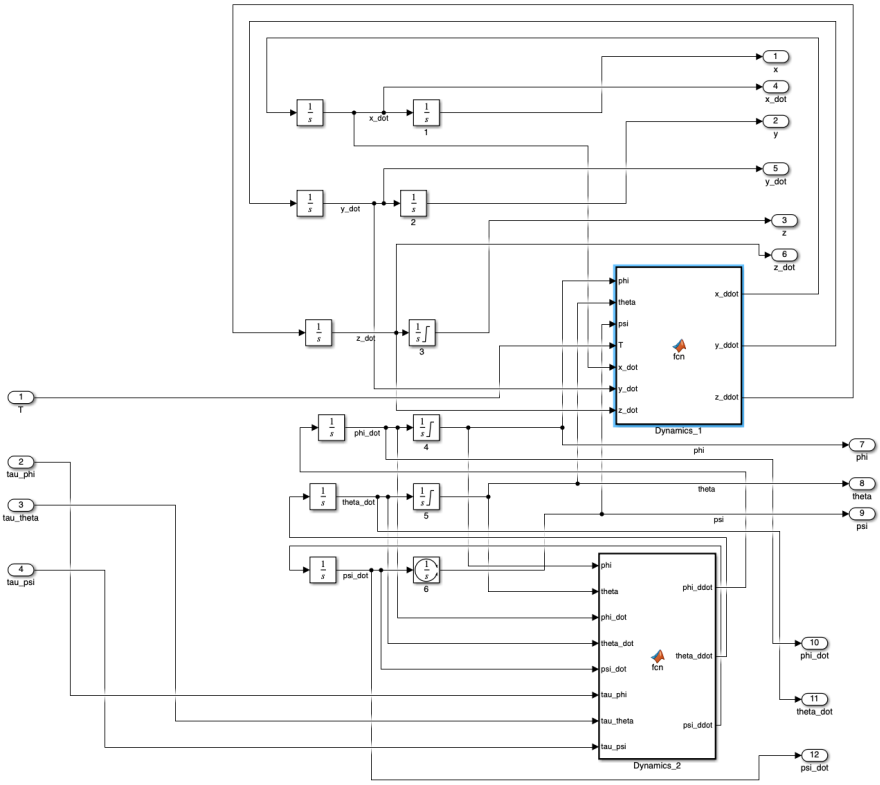


Figure 6: Quadcopter System

This model implements the UAV dynamics simulation, focusing on computing the evolution of **position** and **attitude**. It consists of two main dynamics modules: **Dynamics_1** and **Dynamics_2**, which compute the UAV's **linear motion** and **angular motion**, respectively.

The primary function of `Dynamics_1` is to compute the linear acceleration of a quadcopter in the global coordinate system. It is based on the input attitude angles (ϕ, θ, ψ), thrust (T), and current velocity $(\dot{x}, \dot{y}, \dot{z})$. Considering the effects of gravity, thrust, and aerodynamic drag, the function ultimately outputs the acceleration $(\ddot{x}, \ddot{y}, \ddot{z})$.

The primary function of `Dynamics_2` is to calculate the angular acceleration of a quadcopter in the global coordinate system, i.e., $(\ddot{\phi}, \ddot{\theta}, \ddot{\psi})$ (the angular accelerations around the X, Y, and Z axes of the body-fixed frame).

2.7 Basic Realization

K_p^z, K_i^z, and K_d^z correspond to the PID parameters for Altitude Control.

K_p^{pos}, K_i^{pos}, and K_d^{pos} correspond to the PID parameters for Position Control.

K_p^{att}, K_i^{att}, and K_d^{att} correspond to the PID parameters for Attitude Control.

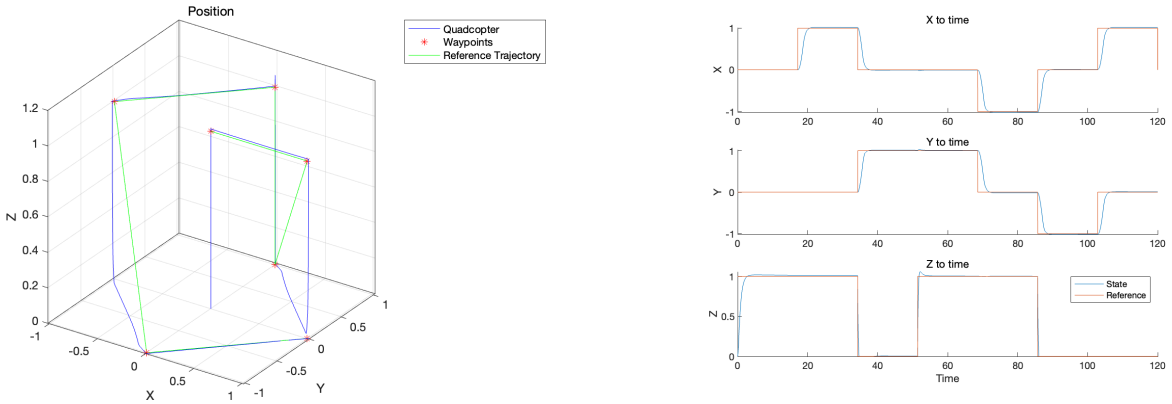
K_p^{ψ}, K_i^{ψ}, and K_d^{ψ} correspond to the PID parameters for Yaw Control. The most suitable combination of parameters is found as follows.

	P	I	D
K*_z	4	0.1	3
K*_pos	6	0.1	4
K*_att	8	0.1	1
K*_psi	8	0.1	1

Table 1: Best combination of PID parameters

The designed trajectory was used for testing, including changes in x, y, and z positions. The flight simulation lasted 120 seconds and the trajectory consisted of 10 waypoints. The experimental results are shown in the figure.

2.8 Test on simple Track



(a) Quadcopter System (view 1)

(b) Quadcopter System (view 2)

Figure 7: Quadcopter System Simulation Results

3 Optimized and More Accurate Dynamic Model

3.1 Overview

In the group project, a more realistic air drag model was implemented to enhance the quadcopter simulation. The traditional linear drag model was replaced with a velocity-dependent quadratic drag model, considering wind influence to better represent real-world flight conditions.

Specifically, the relative wind velocity is calculated as follows:

$$\mathbf{V}_{rel} = \mathbf{V}_{drone} - \mathbf{V}_{wind} \tag{1}$$

Then, the drag force acting on the quadcopter in global coordinates is computed using a quadratic drag equation:

$$\mathbf{D} = \frac{1}{2}\rho_{air} C_d A \left|\mathbf{V}_{rel}\right| \mathbf{V}_{rel} \tag{2}$$

where:

- ρ_{air} is the air density (typically 1.225 kg/m^3 at sea level).
- C_d is the drag coefficient.
- A is the effective frontal area of the drone.

Thus, the translational acceleration of the quadcopter considering drag is calculated by:

$$\ddot{\mathbf{p}} = \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + \frac{1}{m} R_{BG} \begin{bmatrix} 0 \\ 0 \\ T \end{bmatrix} - \frac{1}{m} \mathbf{D}_{vec} \tag{3}$$

where $\mathbf{D}_{vec}$ is the drag force vector computed by:

$$\mathbf{D}_{vec} = \frac{1}{2}\rho_{air} C_d A \left|\mathbf{V}_{rel}\right| \mathbf{V}_{rel} \tag{4}$$

This modification incorporates wind conditions more realistically into the simulation by using relative velocity and quadratic drag, enhancing the fidelity of quadcopter dynamics in varied wind environments.

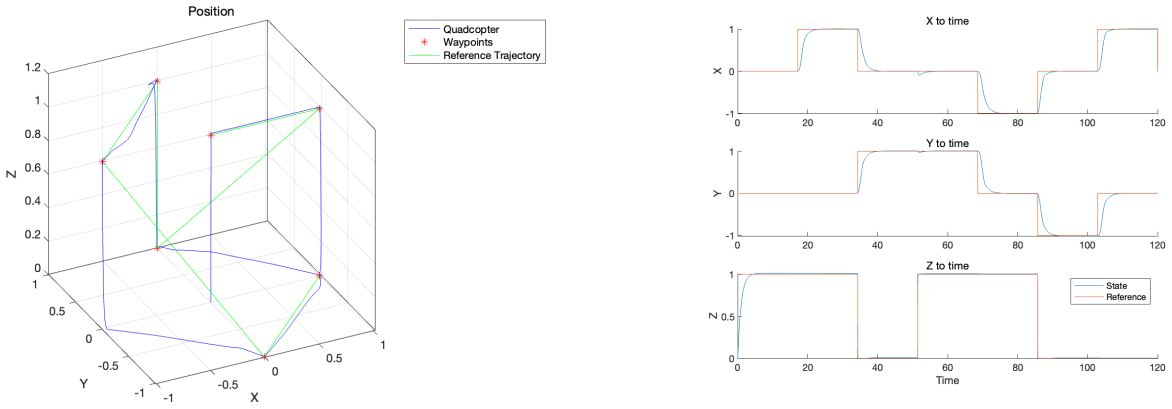
3.2 Experiment Setup

In the experiment, a uniform wind speed of 6 m/s was set along the x-direction. The trajectory used remained consistent with previous experiments. After conducting trials, the best parameter combinations and results were determined, as shown in Table 5.

	P	I	D
K_z	15	0.2	15
K_pos	16	0.1	20
K_att	20	0.5	5
K_psi	15	0.2	8

Table 2: Best combination of PID parameters

3.3 Experiment Results



(a) Quadcopter System - Experiment Result 1

(b) Quadcopter System - Experiment Result 2

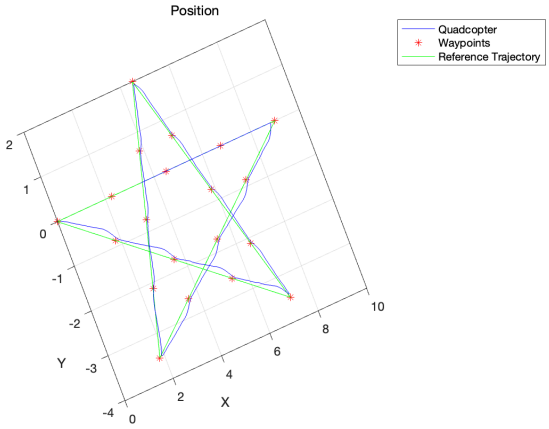
Figure 8: Quadcopter System - Experiment Results

4 [Yang Peilin]: Drawing Complex Trajectories

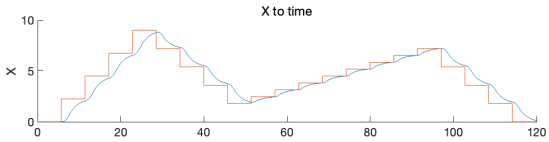
In this part of the experiment, I used the UAV to draw a five-pointed star (pentagram) shape by dividing the trajectory into multiple waypoints. To achieve more accurate path tracking, I tuned the PID parameters shown below:

	P	I	D
K_z	4	0.1	15
K_pos	16	0.1	34
K_att	30	0.1	10
K_psi	30	0.1	10

Table 3: Best combination of PID parameters for the complex pentagram trajectory.



(a) Complex trajectory result (first view).



(b) Complex trajectory result (second view).

Figure 9: Complex trajectory results.

5 [WEI TAO]: Wind Speed Challenges in Different Directions

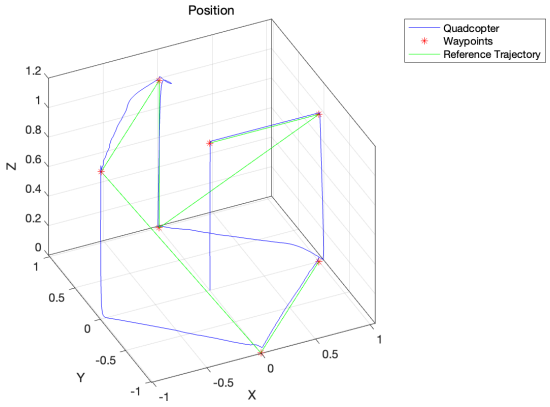
5.1 Effect of Varying Wind Speed

In this section, I set different wind speeds in the x, y, and z directions at different time intervals, covering the early, middle, and late stages of the flight. The goal was to tune the PID parameters to stabilize the UAV under these varying wind conditions.

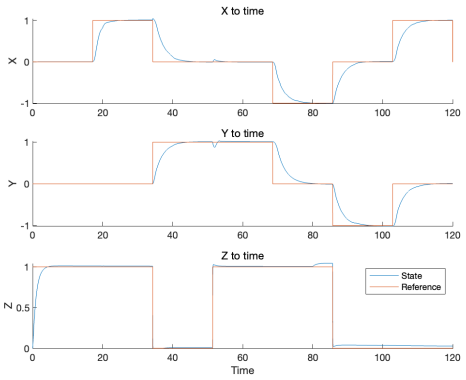
After adjustments, the optimized PID parameters are shown in Table 5.

	P	I	D
K_z	15	0.2	15
K_pos	15	0.12	36
K_att	20	0.2	6
K_psi	16	0.2	8

Table 4: Optimized PID parameters for flight stabilization under varying wind conditions.



(a) Complex trajectory result (first view).



(b) Complex trajectory result (second view).

Figure 10: Complex trajectory results.

6 [XIONG RUIYI]: Multi-UAV Flight

6.1 Trajectory Optimization and Stability Adjustment

In this section, we conducted experiments using two UAVs flying simultaneously. The goal was to record their trajectories while tuning control parameters to achieve more stable flight performance.

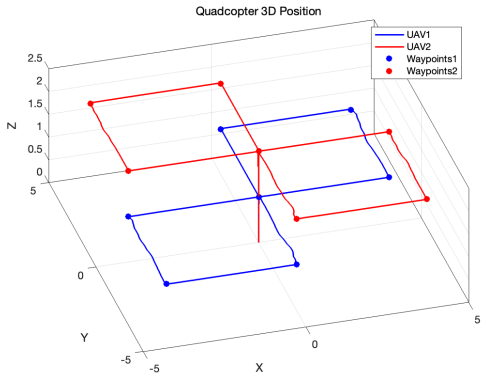
After parameter tuning, the optimized PID values are summarized in Table 5.

	P	I	D
K_z	4	0.1	3
K_pos	6	0.1	4
K_att	8	0.1	1
K_psi	8	0.1	1

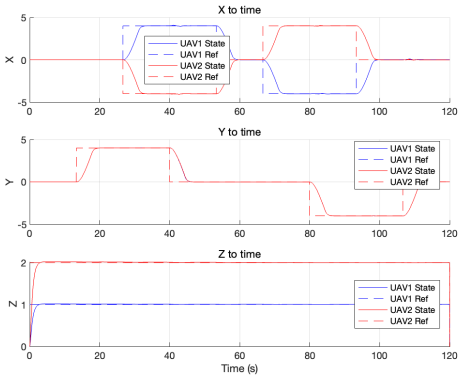
Table 5: Optimized PID parameters for multi-UAV flight stabilization.

6.2 Experimental Results

The following figures illustrate the recorded UAV flight trajectories and their performance under the optimized PID parameters.



(a) Multi-UAV flight experiment results (first view).



(b) Multi-UAV flight experiment results (second view).

Figure 11: Multi-UAV flight experiment results.

6.3 Conclusion

In this experiment, we first analyzed the structure of the Simulink simulation and explained the principles underlying the UAV implementation. We then performed parameter tuning on a moderately complex trajectory to optimize the UAV's flight path. Next, we considered a more complex flight mission by incorporating more realistic aerodynamic drag, successfully completing the group tasks.

In the individual tasks, we successively achieved flight along complex trajectories, implemented multi-UAV flight, and took into account more complex wind-resistance scenarios. At each stage, we performed PID parameter tuning, ultimately completing this experiment.