AE352: Dynamics of a Quadcopter

Zayna Khan
Lead Engineer

Sekeun Oh
Design Engineer

Mo Phoompho
Implementation / Quality Control Engineer

The paper describes the development of a quadcopter model that overcomes the limitations of Euler angles by employing quaternion mathematics to achieve precise attitude control. The model satisfies strict performance and power efficiency criteria since it is designed with six degrees of freedom and changeable rotor speeds. The model's ability to perform intricate, steady maneuvers in real-world scenarios is validated by simulation findings, furthering UAV technology with a scalable and inventive approach to aerial robotics.

I. Nomenclature

A, B, C, D	=	constant matrices of the state-space models
f, g	=	functions of dynamic model and sensor model r
p_x	=	position in the x-direction
p_y	=	position in the y-direction
p_z	=	position in the z-direction
ψ	=	yaw angle
θ	=	pitch angle
ϕ	=	roll angle
v_x	=	linear velocity along the x-axis
v_y	=	linear velocity along the y-axis
v_z	=	linear velocity along the z-axis
w_x	=	angular velocity about the x-axis
w_y	=	angular velocity about the y-axis
w_z	=	angular velocity about the x-axis
f_z	=	net rotor force
J	=	angular velocity transformation matrix
R	=	rotational matrix
τ_x, τ_y, τ_z	=	net torque about x,y and z axes respectively
W_c	=	controllability matrix
W_o	=	observability matrix

II. Introduction

In the realm of aerospace advancement, quadcopters serve as a demonstration of engineering creativity and adaptability. Due to their capacity for vertical takeoffs, landings, and prolonged hovering, quadcopters serve as excellent examples of modern aeronautical engineering. The six degrees of freedom of a quadcopter will be meticulously simulated in our project, highlighting the ability to directly manipulate its state and attitude by making precise adjustments to rotor speed. Our objective is to deliver a robust simulation that not only meets specified engineering requirements but also demonstrates the drone's operational capabilities including steady hovering, maneuverability, and effective energy consumption. This model aims to use quaternion mathematics instead of Euler angles in attitude control to overcome traditional challenges for a more realistic UAV design. This project will enhance our understanding of drone dynamics, paving the way for advanced applications in sectors like defense, delivery systems, and emergency response.

III. Theory

A. Introduction to Quadcopter Dynamics

Quadcopters are flying devices that use four rotors in a symmetrical arrangement to generate lift, enabling them to take off and land vertically and hover accurately. The special arrangement of these rotors allows a quadcopter to move with a great amount of flexibility, exceeding that of traditional aircraft. The fundamental principles of flight physics and mechanics are central to how quadcopters operate. These machines use thrust, lift, drag, and weight to control their stability and motion in the air. The motion of a quadcopter involves both movement and rotation in space and has six degrees of freedom: three directions of movement (forward/backward, up/down, left/right) and three directions of rotation (pitch, roll, yaw). Controlling these movements involves adjusting the rotor speed, which then impacts the vehicle's thrust and torque dynamics. Comprehending these dynamics is essential, as it guides the development of control algorithms that uphold stability and allow for precise control needed for intricate maneuvers. The quadcopter is propelled upwards by the reaction of accelerating air downwards generated by the rotors. A quadcopter can maneuver through three-dimensional space with impressive precision by adjusting the speed of each rotor independently to control tilt, yaw, and roll.

B. Euler angles and Quaternions

Euler angles are frequently utilized in 3D graphics and simulations to indicate the position of objects by describing rotations about the primary axes of pitch, roll, and yaw. This approach is easy to understand and commonly used in computer graphics to animate characters and objects, making it simple to adjust object orientations to create specific outcomes. Nonetheless, Euler angles have their own limitations, particularly the problem of "gimbal lock" when two rotation axes align, causing a reduction in rotational freedom by one degree. This can greatly impede the accuracy of 3D simulations that need free movement, like in aerospace or robotics, where precise orientation is vital. Quaternions provide a strong solution for representing rotations without the possibility of gimbal lock, due to their four-dimensional complex number system. This feature is particularly useful for ongoing, immediate calculations necessary in dynamic simulations and control systems. In our project, employing quaternions for drone attitude control results in more dependable and uniform simulations. Their precision and resistance to numerical inaccuracies allow for accurate control of the drone's position, which greatly improves the accuracy and capability of our 3D simulation system. The rotation dynamics of the Quadrotor in quaternions form is given as :

$$\dot{q} = \frac{1}{2} \begin{bmatrix} 0 \\ \omega \end{bmatrix} \otimes q \tag{1}$$

$$\dot{\omega} = I^{-1} \left(\tau - \omega \times (I\omega) \right) \tag{2}$$

(Use the pdf file Mo sent on Friday to show how euler angles and quaternions can be changed from one another)

C. Attitude control implementation using Quaternions

Euler angles and quaternions serve as mathematical tools for representing 3D orientations and rotations, each with its own set of advantages and limitations. Euler angles, defined by rotations around the axes of a fixed coordinate system, are intuitive and easy to visualize. They are commonly used in many fields, including aerospace, robotics, and computer graphics, due to their simplicity. However, Euler angles suffer from a phenomenon called "gimbal lock," where certain orientations lead to a loss of one degree of freedom, causing ambiguity in the representation and complicating calculations. Gimbal lock can pose significant challenges in applications where precise and continuous control of orientation is required.

In contrast, quaternions offer a more robust and efficient solution for representing 3D rotations. A quaternion is a mathematical construct consisting of a scalar and a three-dimensional vector, capable of representing any rotation in 3D space without encountering gimbal lock. Quaternions provide numerical stability and continuity, making them particularly suitable for applications such as computer graphics rendering, motion planning in robotics, and orientation tracking in virtual reality systems. Despite their mathematical complexity compared to Euler angles, quaternions offer several advantages, including efficient interpolation, uniformity in representing rotations, and straightforward composition of rotations.

The conversion equations between Euler angles and quaternions facilitate seamless transformation between these two representations (Figure 3), allowing developers to leverage the benefits of each as needed. By converting Euler angles to

quaternions, applications can mitigate the risk of gimbal lock and ensure robustness in representing 3D orientations. Conversely, converting quaternions to Euler angles (Figure 4) enables intuitive visualization and interpretation of rotations in terms of familiar pitch, yaw, and roll angles. This interchangeability between Euler angles and quaternions empowers developers to choose the representation that best suits their specific requirements while maintaining the flexibility to switch between them as necessary, ultimately enhancing the efficiency and accuracy of 3D orientation and rotation calculations.

$$\begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} \cos\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right)\cos\left(\frac{\psi}{2}\right) + \sin\left(\frac{\phi}{2}\right)\sin\left(\frac{\theta}{2}\right)\sin\left(\frac{\psi}{2}\right) \\ \sin\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right)\cos\left(\frac{\psi}{2}\right) - \cos\left(\frac{\phi}{2}\right)\sin\left(\frac{\theta}{2}\right)\sin\left(\frac{\psi}{2}\right) \\ \cos\left(\frac{\phi}{2}\right)\sin\left(\frac{\theta}{2}\right)\cos\left(\frac{\psi}{2}\right) + \sin\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right)\sin\left(\frac{\psi}{2}\right) \\ \cos\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right)\sin\left(\frac{\psi}{2}\right) - \sin\left(\frac{\phi}{2}\right)\sin\left(\frac{\theta}{2}\right)\cos\left(\frac{\psi}{2}\right) \end{bmatrix} \tag{3}$$

$$\begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} \text{atan2}\left(2(q_0q_1 + q_2q_3), 1 - 2(q_1^2 + q_2^2)\right) \\ \sin^{-1}\left(2(q_0q_2 - q_3q_1)\right) \\ \text{atan2}\left(2(q_0q_3 + q_1q_2), 1 - 2(q_2^2 + q_3^2)\right) \end{bmatrix} \tag{4}$$

In the context of a drone's orientation, we often refer to two frames: the body frame and the global frame. The body frame is relative to the drone itself, where its axes are aligned with the drone's physical structure. Typically, the X-axis points forward, the Y-axis points to the right, and the Z-axis points downwards (in a right-handed coordinate system). On the other hand, the global frame represents an external reference frame, such as the Earth's coordinate system, where the axes are fixed relative to the environment. The X-axis usually points east, the Y-axis points north, and the Z-axis points upwards.

Quaternions offer significant advantages over Euler angles when it comes to calculating rotations between these frames, especially for drones. They can be represented as $q = q_0 + q_1 i + q_2 j + q_3 k$, where q_0 is the scalar part and q_1, q_2, q_3 are the vector part, and i, j, k are the quaternion imaginary units. Quaternion rotation of a vector v by quaternion q can be expressed as $v' = qvq^*$, where v' is the rotated vector, and q^* denotes the conjugate of q. Quaternion multiplication is non-commutative, defined as $q_1 \otimes q_2 = (s_1 s_2 - \mathbf{v}_1 \cdot \mathbf{v}_2, s_1\mathbf{v}_2 + s_2\mathbf{v}_1 + \mathbf{v}_1 \times \mathbf{v}_2)$, where s_1, s_2 are the scalar parts and $\mathbf{v}_1, \mathbf{v}_2$ are the vector parts of q_1, q_2 respectively.

IV. Design Components

V. Model

A. Requirements

The Engineering Requirements for Quadcopter Model are as follows:

- **ER1: Degrees of Freedom and Dynamics**
  The quadcopter model needs to have six degrees of freedom, covering translational (x, y, z axes) and rotational (pitch, roll, yaw) motions. It must faithfully replicate the forces of gravity and the upward thrust from four rotors.
- **ER2: Control Through Rotor Speed**
  The rotor speeds must be adjusted to control motor torques, enabling the model to change its state and attitude. This allows for exact control and stabilization using either pilot commands or automated flight plans.
- **ER3: Performance and Power Efficiency**
  The quadcopter must achieve designated performance targets without draining its battery too soon. Efficiently managing power consumption is crucial for maintaining long-term operational abilities and completing the planned flight maneuvers.
- **ER4: Realistic and Implementable Design**
  Parameters like weight, energy usage, torque of the engine, and size of the rotor must be practical and sourced from parts that are currently on the market. The drone needs to weigh from 0.1 kg to 10 kg to guarantee practicality and adherence to regulations.

These criteria direct the design of a quadcopter model that is both theoretically sound and practically possible for real-world use.

B. Translational Dynamics

This section addresses the main dynamics affecting the drone: the lift generated by the rotors and gravitational forces. Velocity measurements, crucial for our dynamic model, are derived from sensors mounted directly on the drone's body frame, which is inherently linked to the drone's center of mass (COM) that moves and rotates with the drone. This necessitates a reliable transformation between the drone's body frame and the global reference frame, facilitated by the use of rotation matrices.

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi) & \cos(\phi) \end{bmatrix} \tag{1}$$

$$R_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \tag{2}$$

$$R_z(\psi) = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

To obtain the full transformation from the body frame to the world frame, we employ the Tait-Bryan sequence of yaw-pitch-roll rotations, combining these matrices sequentially:

$$R_{body \to world} = R_z(\psi) R_y(\theta) R_x(\phi) \tag{4}$$

Application of $R_{body \to world}$ enables the transformation of vectors from the body frame to the world frame, facilitating the computation of the drone's motion equations. To extract the position derivatives p_x, p_y, and p_z, we apply this rotation to the body velocity vector:

$$\begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} = R_{body \to world} \cdot \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \tag{5}$$

For the drone's acceleration modeling in the global frame, Newton's second law of motion is adapted to incorporate the force equilibrium:

$$m\dot{r} = F + \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} \tag{6}$$

where r denotes the position vector of the drone's center of mass and F represents the force vector emanating from the rotors in the body frame. The transformation of gravitational forces to the body frame is achieved via the transpose of $R_{body \to world}$:

$$f_{body} = R^T_{body \to world} \cdot \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} \tag{7}$$

In our simplified model, the Coriolis force, which is contingent upon the body's velocity in the rotating frame, is omitted for clarity. This results in the linear acceleration vector being represented as:

$$\dot{v} = f_{body} - \omega \times (m \cdot v) \tag{8}$$

These mathematical formulations are integral for comprehensively understanding the translational dynamics of the drone and ensuring accurate simulations and predictions of its trajectory and stability in flight.

C. Rotational Dynamics

The dynamics of the drone's rotation about its axes are governed by angular velocities and the torques influencing these velocities. To describe these relationships, we utilize the Euler angles: yaw, pitch, and roll, represented by the angular velocity components in the body frame, which can be related to the derivative of the Euler angles through the matrix M.

$$M = \begin{bmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi)/\cos(\theta) & \cos(\phi)/\cos(\theta) \end{bmatrix}$$

This matrix M transforms angular velocities in the body frame into angular rate changes in the Euler angles. The transformation of the angular velocity vector in the body frame can be expressed as:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = M \cdot \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}$$

To determine the drone's angular accelerations, $\dot{\omega}$, we apply Newton's second law of rotation within the body frame. The inertia matrix J, assumed diagonal for simplicity, directly influences the rotational dynamics:

$$J = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}$$

The equation of motion for the angular acceleration, taking into account external torques τ and gyroscopic effects due to rotation, is:

$$J\dot{\omega} + \omega \times (J\omega) = \tau$$

Where $\omega \times (J\omega)$ represents the gyroscopic torque. Solving for $\dot{\omega}$, we get:

$$\dot{\omega} = J^{-1}(\tau - \omega \times (J\omega))$$

This equation encapsulates the dynamics of rotational motion for the drone and is essential for designing control systems that accommodate for both external and internal torques. The robustness of this model allows for precise manipulation of the drone's orientation, crucial for tasks requiring high stability and accuracy. To finalize our dynamic analysis, we've derived essential expressions that define the drone's angular accelerations. These calculations give us:

$$\begin{bmatrix} \dot{\omega}_x \\ \dot{\omega}_y \\ \dot{\omega}_z \end{bmatrix} = J^{-1}(\tau - \omega \times (J\omega))$$

where τ denotes the external torques, ω the angular velocity, and J the inertia tensor. The term $\omega \times (J\omega)$ represents the gyroscopic torque due to the rotational dynamics of the drone.

This structured approach in presenting the final equations of motion allows us to capture the drone's dynamic behavior comprehensively. By establishing a linear model, we pave the way for applying predictive and corrective control mechanisms that ensure the drone's stability and responsiveness in various operating conditions.

D. EOM and linearization

With the complete equations of motion, we linearizing the system to enhance control strategy implementation. Linearization simplifies the model, facilitating more straightforward application of various control techniques. Here:

$$f = \begin{bmatrix} p_x \\ p_y \\ p_z \\ \phi \\ \theta \\ \psi \\ v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} v_z\cos(\theta)\cos(\phi) + v_y\sin(\phi)\cos(\theta) - v_x(\sin(\phi)\sin(\theta)\cos(\phi) - \sin(\phi)\cos(\phi)\sin(\theta)\sin(\phi)) \\ v_z\sin(\phi)\sin(\theta)\sin(\phi) + v_x\cos(\phi)\cos(\theta) - v_y\sin(\theta) + v_y\sin(\phi)\cos(\theta) + v_x\cos(\phi)\cos(\theta) \\ v_z\cos(\phi)\sin(\theta)\tan(\phi) + v_y\sin(\phi) - 2v_z\phi + v_z\omega_z - \frac{981\sin(\phi)\cos(\phi)}{100\cos(\theta)} \\ 2f_z + v_y\omega_z - v_z\omega_y - \frac{981\tan(\theta)}{100} \\ \frac{100000r_z}{33} - \frac{3774\omega_z}{23} \\ 100000\theta + \frac{3774\omega_z}{23} \\ 250\phi \end{bmatrix}$$

and g is, g=$$\begin{bmatrix} p_x + \frac{7\cos(\psi)\cos(\theta)}{40} \\ p_y + \frac{7\sin(\psi)\cos(\theta)}{40} \\ p_z - \frac{7\sin(\theta)}{40} \\ p_x - \frac{7\cos(\psi)\cos(\theta)}{40} \\ p_y - \frac{7\sin(\psi)\cos(\theta)}{40} \\ p_z + \frac{7\sin(\theta)}{40} \\ p_x + \frac{7\sin(\phi)\sin(\theta)\cos(\psi)}{40} - \frac{7\sin(\psi)\cos(\phi)}{40} \\ p_y + \frac{7\sin(\phi)\sin(\psi)\sin(\theta)}{40} + \frac{7\cos(\phi)\cos(\psi)}{40} \\ p_z + \frac{7\sin(\phi)\cos(\theta)}{40} \\ p_x - \frac{7\sin(\phi)\sin(\theta)\cos(\psi)}{40} + \frac{7\sin(\psi)\cos(\phi)}{40} \\ p_y - \frac{7\sin(\phi)\sin(\psi)\sin(\theta)}{40} - \frac{7\cos(\phi)\cos(\psi)}{40} \\ p_z - \frac{7\sin(\phi)\cos(\theta)}{40} \end{bmatrix}$$

Controllability

While control theory is important to the dynamical behavior of the quadcopter, the focus of this report is on the results of the derived Equations of Motion (EOMs). Therefore, control theory will be discussed here, though only briefly.

The quadcopter has 12 states and 4 inputs. The states are the variables that correspond to the drone's existence in space, while the inputs are the values that the drone produces to change its state variables. These are:

States = $p_x, p_y, p_z, \psi, \theta, \phi, v_x, v_y, v_z, \omega_x, \omega_y, \omega_z$

Inputs = $\tau_x, \tau_y, \tau_z, f_x$

The quadcopter is also equipped with four sensors on each quadrotor that track the drone's position in 3D space, defined as:

$o = g(p_x, p_y, p_z, \phi, \theta, \psi)$

The dynamic state of the quadcopter is defined by the state variables and matrices A and B:

$$\dot{x} = Ax + Bu$$

The output of the quadcopter is defined by the sensor equations and matrices C and D:

$$y = Cx + Du$$

After verifying that the system is both controllable, stabilizable, observable, and detectable, we implement a Linear Quadratic Regulator (LQR) to compute the controller gain matrix K. Rewriting our dynamic state using this method yields:

$$\dot{x} = A - BK$$

(1)

Performing LQR again for our observer, we derive L, which is used in our controller. Combining these control methods together yields the controller for the system:

$$u = -K(\hat{x} - x_{\text{des}})$$

where x_{des} is the state the system approaches.

$$\hat{x} = (A\hat{x} + Bu - L(C\hat{x})) \cdot dt$$

This observer equation compensates for state estimation error and updates the value of $\hat{x}$ at every time step, dt, which will help the controller minimize error in its state estimate over time. The controller will, at every time step, be trying to achieve a goal state. In simpler terms, this means that the drone will always be approaching a desired position, orientation, etc. The control theory methods displayed here help the drone accomplish its performance goals utilizing the EOMs.

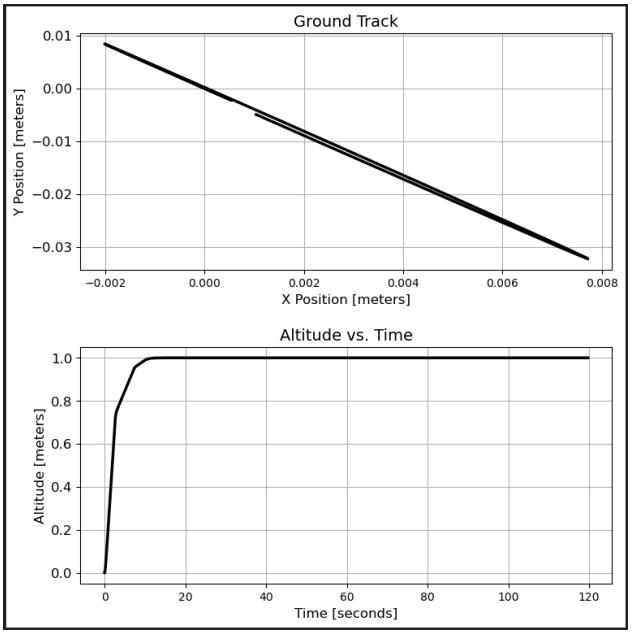

Fig. 1 Fig 1 Hovering Path Graph

VI. Results and Verification

First Performance Goal

The graph shows that the drone was able to maintain a steady hover at 1 meter above the ground for 2 minutes. This stability in altitude over an extended period was achieved through finely tuned PID controllers managing the thrust of each rotor to counteract any unwanted vertical movement or drift. The altitude PID controller effectively adjusted the overall rotor speed to maintain the required hover altitude, demonstrating the system's ability to counteract gravitational forces and any environmental disturbances. The graph can be seen in fig 1.

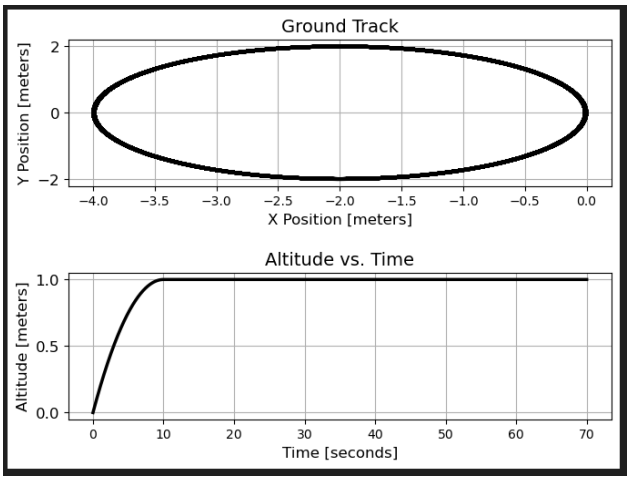

Fig. 2 Fig 2 Circular Path Graph

Second Performance Goal

For the task of flying in a circle with a radius of 2 meters at an altitude of 1 meter, our PID controllers were configured to manage both lateral and vertical stability. The roll and pitch PID controllers played critical roles in maintaining the drone's orientation, allowing it to navigate the circular path smoothly. By continuously adjusting the differential thrust among the rotors, the drone could sustain the necessary lateral motion to trace the circular trajectory, while the altitude PID controller ensured it remained at the correct height throughout the maneuver. The graph can be seen in fig 2.

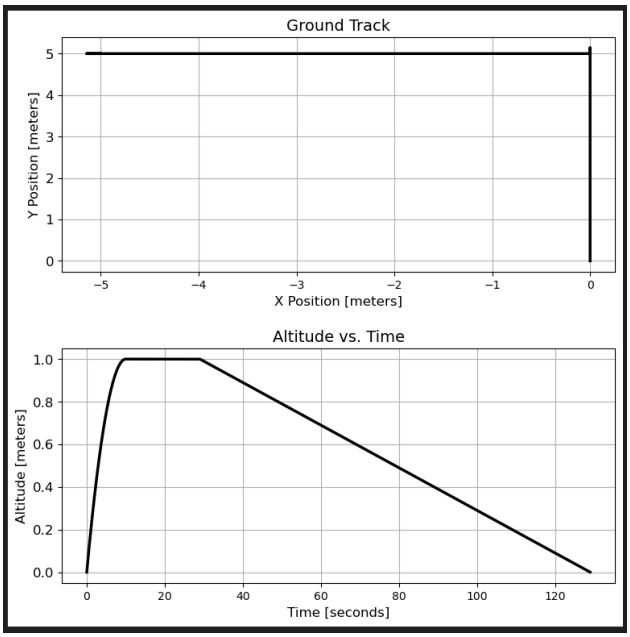

Fig. 3 Fig 3 Maneuvering Path Graph

Third Performance Goal

The complex maneuver involved multiple stages: vertical takeoff, horizontal movement, a yaw turn, and a controlled landing. Initially, the altitude PID controller managed a stable ascent to 1 meter. Subsequently, roll or pitch controllers, depending on the direction of travel, were used to maintain a steady forward speed as per the trajectory requirements. The yaw PID controller was crucial for executing the 90-degree turn, adjusting the relative rotor speeds to rotate the drone precisely. Finally, the descent was controlled delicately by the altitude PID to land at a safe speed. This multi-stage performance showcases our PID system's capability to handle various flight dynamics seamlessly. The graph can be seen in fig 3.

VII. Conclusion

In conclusion, our quadrotor project successfully demonstrated robust control and precise navigation capabilities, achieving all the outlined performance goals through the integration of sophisticated control theory and practical engineering. The first performance goal, maintaining a stable hover at 1 meter for two minutes, was accomplished by finely tuning the PID controllers to manage the thrust of each rotor, ensuring a consistent altitude despite external disturbances. This required precise adjustments to the drone's rotor speeds, which were continuously updated based on feedback from the drone's altitude sensors.

For the second and third performance goals, our implementation of PID control allowed for complex path following and dynamic maneuvers, respectively. The drone flawlessly followed a circular path with a set radius of 2 meters at an altitude of 1 meter, where the roll and pitch PID controllers were instrumental in maintaining the correct trajectory and altitude. The third goal involved a sequence of maneuvers including vertical takeoff, linear motion, a yaw turn, and a gentle landing, which were executed through coordinated control over all four rotors. This complex sequence not only demonstrated the drone's agile maneuvering capabilities but also highlighted the efficiency of our control algorithms in real-time adjustment and error minimization. Overall, the project not only met its technical objectives but also provided valuable insights into the practical applications of PID controllers and the integration of feedback mechanisms in unmanned aerial vehicle systems.

VIII. Contributions

A. Contribution by Sekeun Oh

Designated path and quaternion rotational dynamics as well as implementing PID controls to control the movements of the drone.

B. Contribution by Zayna

-Prepared technical documentation, ensuring clarity and precision.

C. Contribution by Mo

Derived equations of motion and implemented code for the drone acheiving the first performance goal

IX. References

1. PDF for 2302.03500. Available at: https://arxiv.org/pdf/2302.03500.pdf (Accessed: 07 May 2024).
2. Selby, W. (no date) Quadrotor system modeling - non-linear equations of motion, Wil Selby. Available at: https://wilselby.com/research/arducopter/modeling/ (Accessed: 21 April 2024).
3. https://www.diva-portal.org/smash/get/diva2:1010947/FULLTEXT01.pdf
4. https://github.com/sekeuno2/ae352-final-project