Dynamic Model of a Quadcopter

Amber Miller, Patryk Wojtkowski, and Ishaan Kandamuri

Champaign, Illinois, 61820, United States

I. Nomenclature

x	=	x-axis displacement
y	=	y-axis displacement
z	=	z-axis displacement
θ	=	pitch
ϕ	=	roll
ψ	=	yaw
ω	=	angular velocity
$\dot{\omega}$	=	angular acceleration
I	=	inertia
τ	=	torque
T	=	thrust force
g	=	acceleration due to gravity

II. Introduction and Background

Quadcopters, also known as quadrotors or drones, are unmanned aerial vehicles (UAVs) that utilize four rotors for lift and maneuverability. Their distinct design with four propellers arranged in a cross formation allows them to achieve vertical take-off and landing (VTOL) and exceptional hovering capabilities. This makes them highly versatile for various applications, from photography and videography over search and rescue missions to defense. They are popular due to their ease of control and come in a range of sizes and functionalities.

This report details the dynamical model of a quadcopter that was created to fulfill certain engineering requirements and these three performance goals:

1. Hover 1 meter above ground for 2 minutes.
2. Fly in a circle of radius 2 m, at an altitude of 1 m above ground at a speed of 0.5 m/s for at least 1 minute.
3. Launch from ground and ascend vertically until 1 m above ground. Move in a straight line 1 m above ground at an average speed of 1 m/s for 5 m, stop (hover), yaw 90 degrees to the left, move in another straight line for 5 m, stop (hover), and land vertically with a speed of no more than 1 cm/s.

Furthermore, the engineering requirements the quadcopter must achieve are listed below:

- The drone model shall have 6 degrees of freedom (center of mass motion plus attitude) and account for gravity and lift generated from 4 rotors.
- The drone shall be able to modify its state and attitude solely through increasing or decreasing the number of revolutions per second of its 4 rotor blades, as determined by the motor torques.
- The modeled drone shall be able to achieve the performance goals as outlined above, without exhausting its power supply.
- The model determined through its parameters such as mass, power consumption, engine torques and rotor blade size shall be realistic and implementable, preferably through off-the-shelf parts. In particular, the mass of the drone shall be between 0.1kg and 10kg.

These engineering requirements and performance goals are fulfilled and detailed in the following sections.

III. EOM Derivation

To develop the equations of motion for a quadcopter, an understanding of the required degrees of freedom must first be developed. The quadcopter must have six degrees of freedom; the position states x, y, z, as well as the rotational states ϕ, θ, and ψ, which describe the attitude of the drone. Therefore, we will need two sets of equations of motion; the first set should describe the translational dynamics of the drone, while the other set describes the drone's rotational dynamics.

A. Translational Dynamics

With the degrees of freedom clearly defined, the equations of motion can now be derived. We will start with the translational equations for the x, y, and z degrees of freedom. These can be derived using Newton's Laws in body and inertial frames. To begin, the rotation matrices to move from the inertial frame to the drone body frame must be defined. To accomplish this, we will use the rotation conventions for the 3-2-1 Tait-Bryan angles, where ψ is the yaw angle, θ is pitch, and ϕ is roll. This convention will be used for the rest of the derivations.

For the above convention, the rotation matrices that transfer a vector from the inertial frame to the rotated body frame are as follows.

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

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

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

These matrices represent a series of 3 rotations to move from the inertial frame to the body frame. We can multiply these matrices together to get a single rotation matrix that moves a vector from the inertial frame directly to the body frame, without any steps in between. The required multiplication is:

$$R = R_x(\phi)R_y(\theta)R_z(\psi)$$

$$= \begin{bmatrix} \cos(\psi)\cos(\theta) & \sin(\psi)\cos(\theta) & -\sin(\theta) \\ -\sin(\psi)\cos(\phi) + \cos(\psi)\sin(\theta)\sin(\phi) & \cos(\phi)\cos(\psi) + \sin(\phi)\sin(\psi)\sin(\theta) & \cos(\theta)\sin(\phi) \\ \sin(\psi)\sin(\phi) + \cos(\phi)\sin(\theta)\cos(\psi) & -\cos(\psi)\sin(\phi) + \sin(\psi)\sin(\theta)\cos(\phi) & \cos(\theta)\cos(\phi) \end{bmatrix}$$

However, in these derivations, it will be far more important to move vectors from the body frame to the inertial frame. To do this, we use the orthogonality of rotation matrices to write the following.

$$R^{-1} = R^T$$

$$= \begin{bmatrix} \cos(\psi)\cos(\theta) & -\sin(\psi)\cos(\phi) + \cos(\psi)\sin(\theta)\sin(\phi) & \sin(\psi)\sin(\phi) + \cos(\phi)\sin(\theta)\cos(\psi) \\ \sin(\psi)\cos(\theta) & \cos(\phi)\cos(\psi) + \sin(\phi)\sin(\psi)\sin(\theta) & -\cos(\psi)\sin(\phi) + \sin(\psi)\sin(\theta)\cos(\phi) \\ -\sin(\theta) & \cos(\theta)\sin(\phi) & \cos(\theta)\cos(\phi) \end{bmatrix}$$

R^{-1} is the matrix that allows us to move from the body frame to the inertial frame. With this important formula developed, we can now use Newton's Laws to find the translational equations of motion.

First, we use Newton's Second Law to write the forces acting in the quadcopter body frame. We ignore gravity for now, as it will be accounted for when writing the equations in the inertial frame. These equations are,

$$m\ddot{z} = T_1 + T_2 + T_3 + T_4 = T$$

$$\ddot{x} = \ddot{y} = 0$$

$$\overrightarrow{F_{T,b}} = \begin{bmatrix} 0 \\ 0 \\ T \end{bmatrix}$$

where m is the drone's mass, T_i is the thrust generated by the ith propeller, and b refers to the body frame. In the drone's inertial frame,

$$\overrightarrow{F_{g,I}} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix}$$

$$m\ddot{\vec{q}} = R^{-1}\overrightarrow{F_{T,b}} + \overrightarrow{F_{g,I}}$$

where g is the acceleration due to gravity, and $\vec{q}$ is the drone's position vector. From the above equation, the translational equations of motion are defined as follows:

$$\begin{bmatrix} \ddot{X} \\ \ddot{Y} \\ \ddot{Z} \end{bmatrix} = \frac{1}{m}\begin{bmatrix} T(sin(\phi)sin(\psi) + \cos(\phi)\sin(\theta)\cos(\psi)) \\ T(\sin(\psi)\sin(\theta)\cos(\phi) - \cos(\psi)\sin(\phi)) \\ T\cos(\theta)\cos(\phi) - mg \end{bmatrix}$$

To calculate thrust, the relations between the drone's propellers and torques were used to derive the following equations of motion,

$$T = C_L\omega^2 \text{ and } \tau = C_D\omega^2 + I\dot{\omega}$$

where C_L and C_D are the propeller coefficients of lift and drag respectively, ω is the propeller's angular velocity, and I is the mass moment of inertia of the propeller.

In the context of this dynamical model, the propeller angular acceleration is negligible as we assume instantaneous control of the motors and exclude cases when the propellers change their angular velocity over time, such as during a takeoff or landing, which makes the second term in the τ expression drop out. Therefore, the relationship between torque and thrust can be written as

$$T = \frac{C_L}{C_D}\tau$$

This allows us to rewrite the translational equations in terms of the propeller motor torques.

$$\begin{bmatrix} \ddot{X} \\ \ddot{Y} \\ \ddot{Z} \end{bmatrix} = \frac{C_L}{C_D m}(\tau_1 + \tau_2 + \tau_3 + \tau_4)\begin{bmatrix} sin(\phi)sin(\psi) + \cos(\phi)\sin(\theta)\cos(\psi) \\ \sin(\psi)\sin(\theta)\cos(\phi) - \cos(\psi)\sin(\phi) \\ \cos(\theta)\cos(\phi) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix}$$

B. Rotational Dynamics

To obtain the rotational equations of motion, Euler's equations of motion for rigid bodies will be very helpful. These equations are written as,

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

where I is the moment of inertia tensor, ω is the body frame angular velocity, and τ is the total torque vector in roll, pitch, and yaw. Since our body frame is defined along the drone's rotational axes with a symmetrical body, we define I and τ as:

$$I = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \text{ and } \tau = \begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix}$$

Solving for $\dot{\omega}$, we get the body frame angular accelerations of the quadcopter.

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

$$= \begin{bmatrix} \frac{1}{I_{xx}} & 0 & 0 \\ 0 & \frac{1}{I_{yy}} & 0 \\ 0 & 0 & \frac{1}{I_{zz}} \end{bmatrix} \left(\begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} - \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \times \begin{bmatrix} I_{xx}\omega_x \\ I_{yy}\omega_y \\ I_{zz}\omega_z \end{bmatrix} \right)$$

$$= \begin{bmatrix} \dfrac{1}{I_{xx}} & 0 & 0 \\ 0 & \dfrac{1}{I_{yy}} & 0 \\ 0 & 0 & \dfrac{1}{I_{zz}} \end{bmatrix} \left(\begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} - \begin{bmatrix} (I_{zz} - I_{yy})\omega_z\omega_y \\ (-I_{zz} + I_{xx})\omega_z\omega_x \\ (I_{yy} - I_{xx})\omega_x\omega_y \end{bmatrix} \right)$$

$$= \begin{bmatrix} \dfrac{\tau_\phi + (-I_{zz} + I_{yy})\omega_z\omega_y}{I_{xx}} \\ \dfrac{\tau_\theta + (I_{zz} - I_{xx})\omega_z\omega_x}{I_{yy}} \\ \dfrac{\tau_\psi + (-I_{yy} + I_{xx})\omega_x\omega_y}{I_{zz}} \end{bmatrix}$$

The moment of inertia of the drone about each axis is constant and depends on the geometric properties of the drone, so they do not need to be further addressed. However, the total torques about each axis need to be expressed in terms of the four motor torque values. The torque equations in the body frame can be written as below, where l represents the length of the drone arm from the center of mass to the center of each propeller, and τ_i represents the motor torque input of the ith propeller.

$$\tau_\phi = lT_1 - lT_3 = l\frac{C_L}{C_D}(\tau_1 - \tau_3)$$
$$\tau_\theta = lT_2 - lT_4 = l\frac{C_L}{C_D}(\tau_2 - \tau_4)$$
$$\tau_\psi = \tau_1 - \tau_2 + \tau_3 - \tau_4$$

Each equation represents a torque acting about the axis of yaw, pitch, or roll; these axes are equivalent to the x, y, and z body axes. From these torque equations, the final equations of angular acceleration are defined below.

$$\dot{\omega}_1 = \frac{1}{I_{xx}}\left(l\frac{C_L}{C_D}(\tau_1 - \tau_3) + \left(-I_{zz} + I_{yy}\right)\omega_z\omega_y\right)$$
$$\dot{\omega}_2 = \frac{1}{I_{yy}}\left(l\frac{C_L}{C_D}(\tau_2 - \tau_4) + (I_{zz} - I_{xx})\omega_z\omega_x\right)$$
$$\dot{\omega}_3 = \frac{1}{I_{zz}}\left((\tau_1 - \tau_2 + \tau_3 - \tau_4) + \left(-I_{yy} + I_{xx}\right)\omega_x\omega_y\right)$$

Additionally, an equation that relates the angular velocities to the rate of change of the Tait-Bryan angles needs to be defined. A formulation already exists for this relationship, and is written as:

$$\begin{bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{bmatrix} = \begin{bmatrix}0 & \frac{\sin(\psi)}{\cos(\theta)} & \frac{\cos(\psi)}{\cos(\theta)}\\ 0 & \cos(\psi) & -\sin(\psi)\\ 1 & \tan(\theta)\sin(\psi) & \tan(\theta)\cos(\psi)\end{bmatrix}\begin{bmatrix}\omega_1\\ \omega_2\\ \omega_3\end{bmatrix}$$

When considering alongside the equations of angular acceleration, we get two systems of differential equations describing the rotational motion of the quadcopter. When solved, these ODEs allow us to fully understand the quadcopter's attitude over time, and as such, are our final equations of rotational motion.

IV. Control Methods

To determine the input torques necessary to control the trajectory of our drone, we modeled our EOM in state-space form. By organizing our EOM in a dummy matrix, $\dot{x}$, where x are the inputs in system, defined as

$$[x] = \begin{bmatrix}x\\ y\\ z\\ \psi\\ \theta\\ \varphi\\ \dot{x}\\ \dot{y}\\ \dot{z}\\ \dot{\varphi}\\ \dot{\theta}\\ \dot{\varphi}\end{bmatrix}$$

Additionally, we can model our input torques as u, defined as,

$$[u] = \begin{bmatrix}\tau_1\\ \tau_2\\ \tau_3\\ \tau_4\end{bmatrix}$$

where $\tau_1, \tau_2, \tau_3, \tau_4$ are the input torques on each of the four propellers across the drone.

Given our equation of motion is a function of the system states and inputs, x and u, we can call our EOM $f(x,u)$. We can now linearize our EOM $f(x,u)$ about an equilibrium point, and then reorganize our EOM into state-space form, where

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

and A and B are the Jacobian of $f(x,u)$ with respect to x and u. For our equilibrium point, we chose,

$$[x_0] = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$[u_0] = \begin{bmatrix} .25 \cdot m^2 \cdot \frac{g}{C} \\ .25 \cdot m^2 \cdot \frac{g}{C} \\ .25 \cdot m^2 \cdot \frac{g}{C} \\ .25 \cdot m^2 \cdot \frac{g}{C} \end{bmatrix}$$

where C = C_L/C_D. Once we have our EOM represented in state space form, we can solve for the inputs necessary to achieve our engineering requirements by modeling an optimal controller using LQR. By varying the cost functions, Q and R for each of the three tasks, alongside the controller initial conditions, the drone is able to vary its trajectory accordingly.

V. Results

In order to determine the parameters of our specific drone using the parts specified above, we modeled a crude approximation of our drone using known dimensions and approximated masses based on material estimates. The resulting NX model was used to determine various parameters used in the equations of motion of our system, including the moments of inertia of the drone body, the moments of inertia of the motor-propeller body, the total mass, and the center of mass position. A screencap of the NX model is shown below.

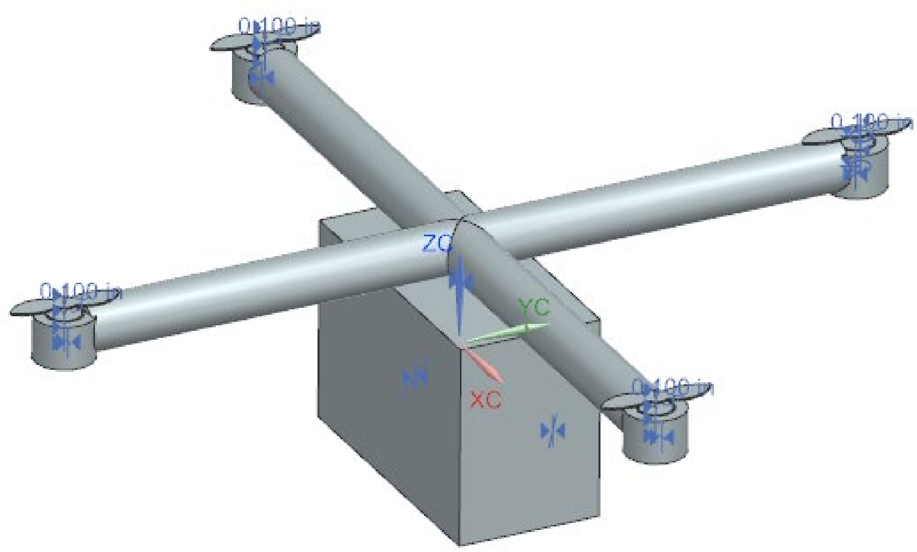

Figure 1. NX model of the quadcopter.

Parameters From NX Model and Software:
m = 0.4990514 kg
I_{xx} = .0004635295 $kg \cdot m^2$
I_{yy} = .0006227343 $kg \cdot m^2$
I_{zz} = .007210787 $kg \cdot m^2$
l = 0.1188466 m

Using LQR to model the optimal controller which will manipulate our input torques and forces in order to bring our system to equilibrium (defined locally), we can redefine our inputs, u, as $u = -Kx$, where K is our optimal controller. Next, we simulate the system using RK45 to solve our IVP and create 3 unique functions which define our $\dot{x}$, one for each test case. Each test case is designed to meet one of the performance goals. The exact details of each test case can be found in the GitHub repository Python file, titled ‘code’, alongside our varying cost matrices. The results are seen below.

Test Case 1

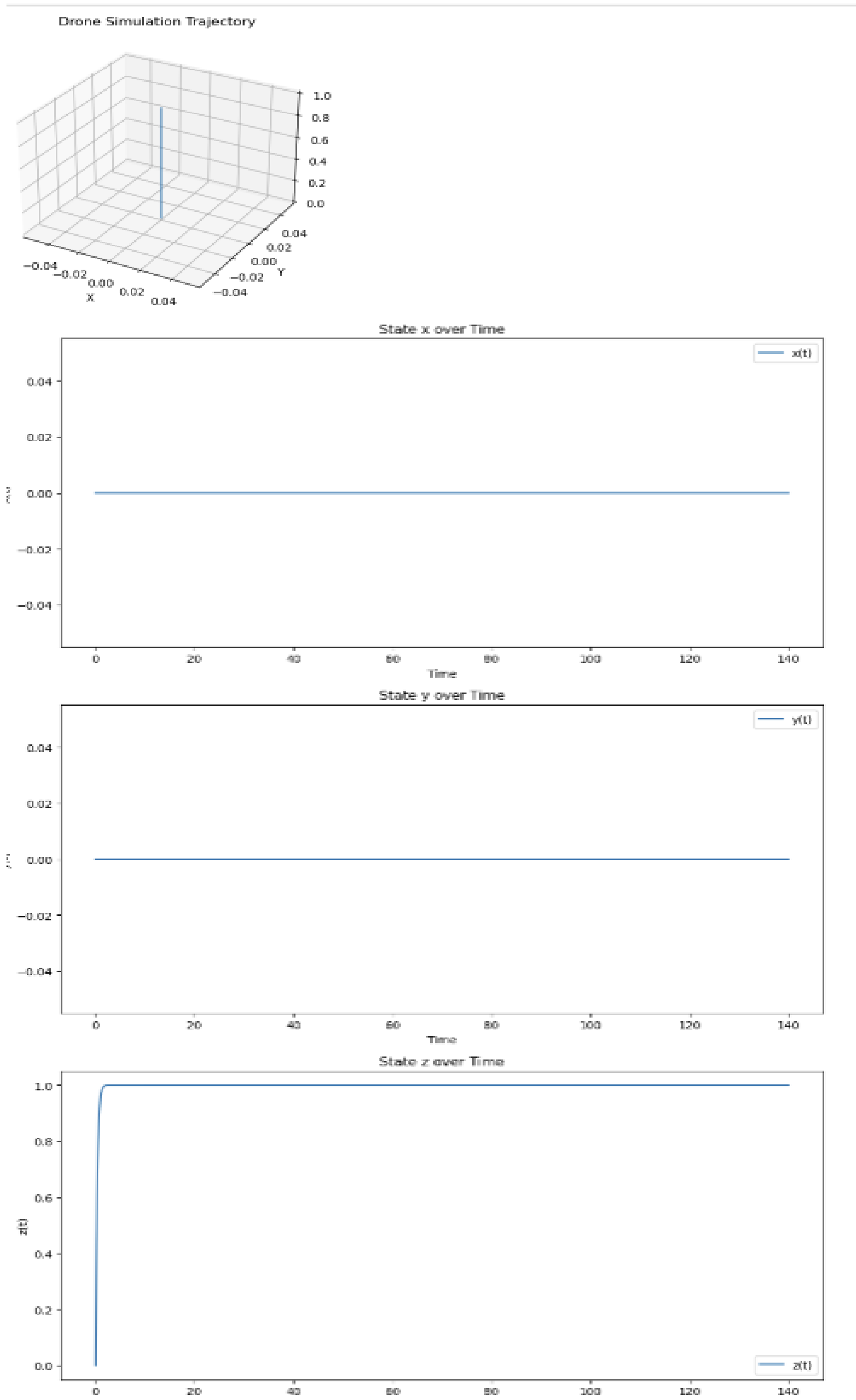

Figure 2. State variables over time for Test Case 1

As seen in Figure 2, the drone hovers 1 m off the ground for over 2 minutes, represented by the state z over time, thus fulfilling the requirements of Performance Goal 1.

Test Case 2

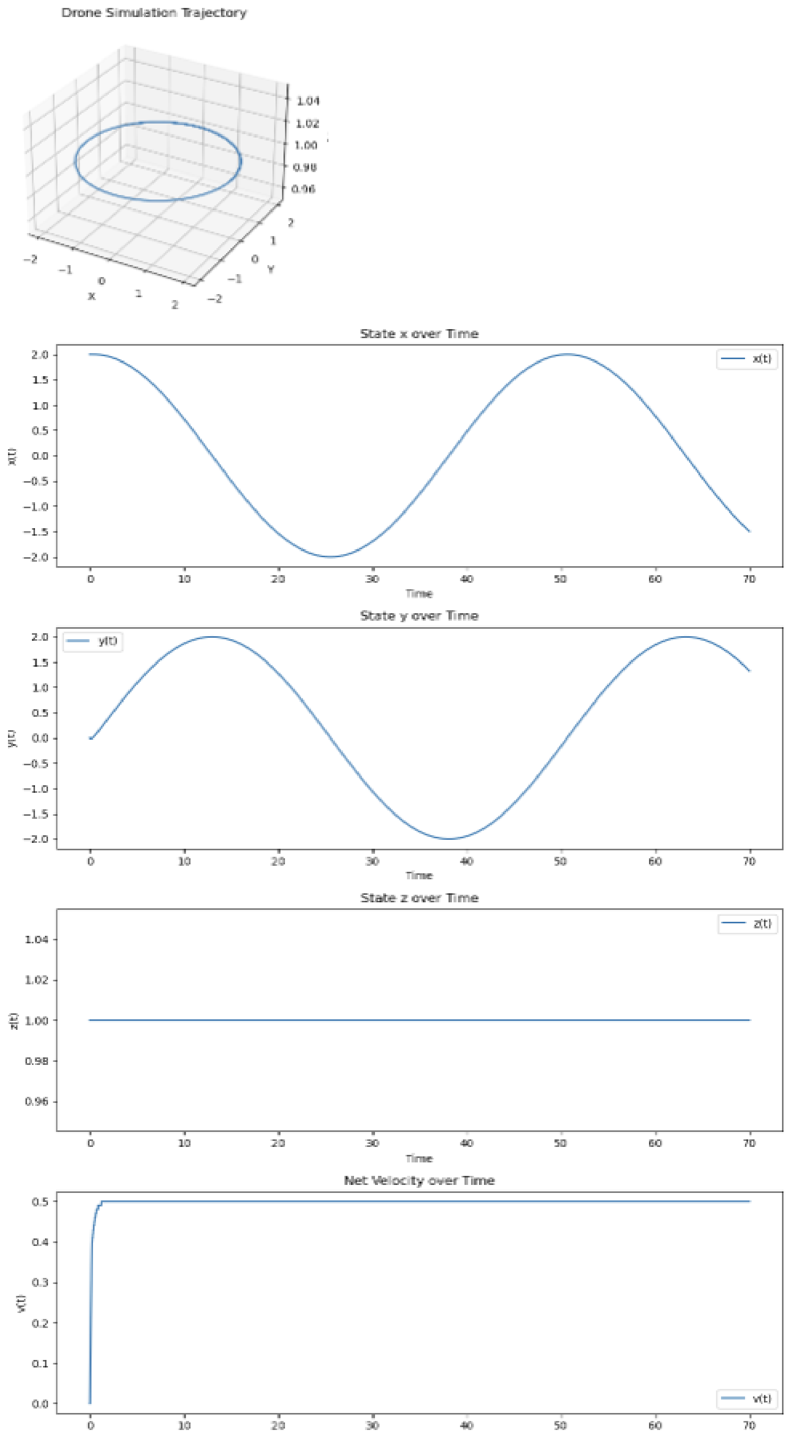

Figure 3. State variables over time for Test Case 2

As seen in Figure 3, the drone flies in a circle with a radius of 2 meters, at an altitude of 1 meter above ground at a speed of 0.5 m/s for at least 1 minute, fulfilling the requirements set by Performance Goal 2.

Test Case 3

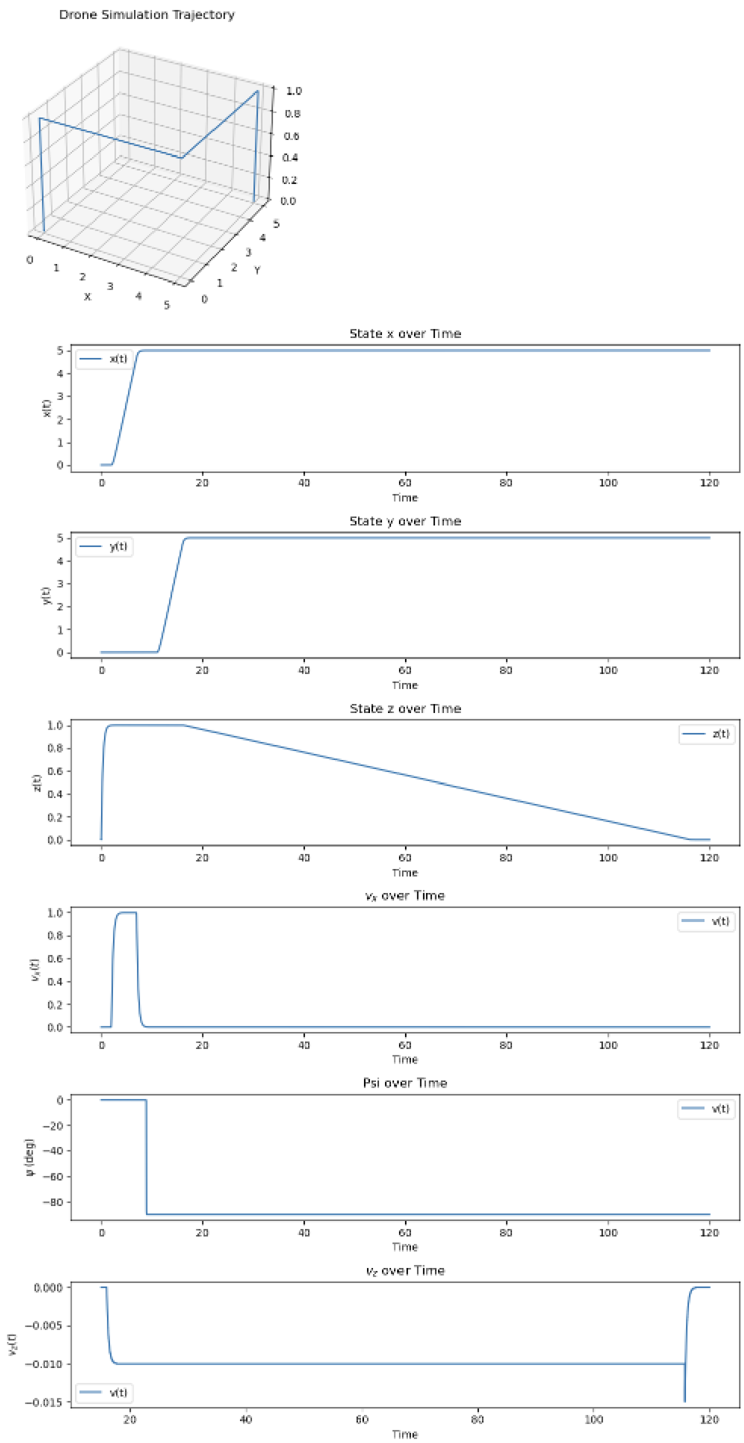

Figure 4. State variables over time for Test Case 3

As seen in Figure 4, the drone launches from the ground and ascends vertically until 1 meter above ground. It then moves in a straight line 1 meter above the ground at an average speed of 1 m/s for 5 m. The drone then yaws 90 deg to the left, moves in another straight line for 5 m, and lands vertically with a speed of no more than 1 cm/s. These maneuvers fulfill the requirements set by Performance Goal 3.

VI. Conclusion

In this report, a dynamical model of a quadcopter has been successfully developed using a combination of Newton's Laws and Euler's equations of motion for rigid, rotating bodies. A control method using LQR was then used to implement the equations of motion so they could be tested. After determining appropriate constants, such as the moments of inertia, as well as lift and drag coefficients for the propellers, the quadcopter's performance was tested in three different scenarios. The quadcopter successfully completed each performance goal, proving that the derived equations of motion were correct.

Additionally, a list of off-the-shelf parts has been developed that could be used to turn this quadcopter model into reality. These parts are generally consistent with the geometry of the model developed in NX, aside from some simplifications seen in Figure 1. The parts are listed below.

Motors	BetaFPV 0702 30000Kv Dual Ball Bearing Micro Motor (4 Pack)	Link
Propellers	Gemfan Durable 1210 Bi-Blade 31mm Propellers	Link
Frame	NewBeeDrone Turismo 5" Lightweight Frame	Link
Battery	Gaoneng GNB 22.2V 6S 1550mAh 100C LiPo Battery	Link
Controller	SpeedyBee F405 V4 Flight Controller - 30x30	Link
Sensors	Matek Optical Flow & Lidar Sensor 3901-L0X 3901 PMW3901	Link

Table 4. List of Components Used to Model the Quadcopter

Contributions

Ash Miller – Code/Control Methods/NX Model/Results

Patryk Wojtkowski – Derivation of Equations of Motion, Written Report

Ishaan Kandamuri – Written Report, Equations of Motion, Ideology

Appendix

Here is the GitHub repository containing our code: https://github.com/amburner/AE352

References

[1] 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).

[2] Lawless, A. (no date) *Flight Test Engineering Reference Handbook*. Available at: https://society-of-flight-test-engineers.github.io/handbook-2013/index.html (Accessed 1 May 2024).