ME-GY 6923 Simulation Tools for Mechatronics and Robotics

Fall 2021

Final Project

Quadcopter Simulation and Altitude Control

DEPARTMENT OF MECHANICAL AND AEROSPACE ENGINEERING

NEW YORK UNIVERSITY

Students Name: Xinzhou Jiang (xj2106)

Rishi Eswar Varma Vegesna (rv2210)

Penmetsa Tejaswi Raju (pr2258)

Arjun Raja (ar6841)

Instructor: Prof. V. Kopman

Table of Content

Abstract

During ME-GY 6923 Simulation Tools for Mechatronics and Robotics Fall 2021 semester, we learned how to use simulation software (MATLAB) to model and control a dynamic system. The report uses MATLAB R2021b Simulink and Simscape (Aerospace Toolbox) to simulate and control a quadcopter in 3D. The design objectives are altitude control, attitude control, and wind disturbance rejection. A brief introduction to MATLAB, Simulink, and Simscape are provided, dynamics of the quadcopter, wind disturbance modeling, trajectory generation, and attitude control methodology are discussed. A short literature review on the quadcopter and wind disturbance is given. Detailed Simulink and Simscape implementations of the importing quadcopter model, design of trajectory generator, design of altitude controller, design of attitude controller, and design of wind disturbance control are included. Control architectures logics are demonstrated with the aid of a flowchart. Results of 3D simulation and roll pitch yaw angle variations concerning time are demonstrated. Analysis, discussion, and conclusion based on the testing results were elaborated. The quadcopter is successfully simulated and controlled with high performance. The valuable programming skill is gained, relevant theories are learned, and practical problem-solving ability is developed.

Keywords: Simulink, MATLAB R2021b, Simscape, Aerospace Toolbox, quadcopter, dynamic, attitude control, altitude control, wind disturbance, trajectory generation, PID control.

Introduction

1.1 Project Outline

A quadcopter includes four symmetrical rotor-propellers positioned at the same height on each side, creating a cross with the same radius from its center, as the name indicates. [1] This report intends to design a quadcopter simulation and control program using MATLAB Simulink R2021b. The quadcopter will perform wind disturbance rejection and altitude and attitude maintenance. PID controller is designed for the altitude control loop and the quadcopter's roll, pitch, and yaw control loops. The final output of the altitude controller is four rpm motors speeds calculated by the computed roll, pitch, yaw, and throttle results. The thrust of each motor will be calculated by the motor rpm and thrust coefficient calculator. PID control blocks are implemented in roll, pitch, yaw movement, and altitude feedback control loop.

The program consists of two parts: Control and dynamics. The control part calculates the dynamics input for each quadcopter's motor based on the desired location. The physical model is referenced by Math Works package delivery quadcopter [6] and it is imported and controlled in the dynamics part. The detailed explanation and formulas will be present in the system model section.

The quadcopter 3D simulation is successfully achieved and the designed PID controllers are working perfectly well. Demonstration of attitude balance and wind disturbance rejection are provided. The scopes of the propeller's thrust and speed, and the quadcopter's attitude and position are provided. The overall performance meets the expectation. Simulation results, pictures, detailed analysis, and discussion are shown in the result and discussion section.

1.2 Working Principle of Quadcopter

The four motors of the quadcopter are arranged in a cross shape, driving the four propellers to rotate to generate upward thrust. The four motor wheels are at the same distance from the geometric center. When the lift generated by the two diagonal shafts is the same, the balance of the torque can be ensured. One pair of reversal motors and another pair of co-rotation motors balance the anti-torque of the rotation around the vertical axis and ensure the stability of the four-axis heading.

1.3 MATLAB, Simulink, Simscape

MATLAB is a programming language that engineers and scientists use to study and build systems and products that change the world. The MATLAB language, a matrix-based language that allows the most natural representation of computational mathematics, lies at the heart of MATLAB. [7]

Simulink is a graphical interface for modeling, simulation, and analysis of dynamic systems. It is a MATLAB add-on product. It allows for the quick development of virtual prototypes that may be used to explore design concepts at any degree of detail with minimum effort. Simulink has a graphical user interface (GUI) for creating models in the form of block diagrams. It comes with a large library of prefabricated blocks that may be used to create graphical representations of systems by dragging and dropping them. The user may create an "up-and-running" model that would normally take hours to construct in a laboratory setting. It can represent linear and nonlinear systems in continuous time, sampled time, or a combination of both. [8]

Simscape is a tool that lets users quickly develop physical system models in the Simulink environment. Simscape allows the creation of physical component models based on physical connections that work with block diagrams and other modeling paradigms. By integrating key components into a schematic, users may simulate systems like electric motors, bridge rectifiers, hydraulic actuators, and refrigeration systems. More complicated components and analytical capabilities are available with Simscape add-on packages. [9]

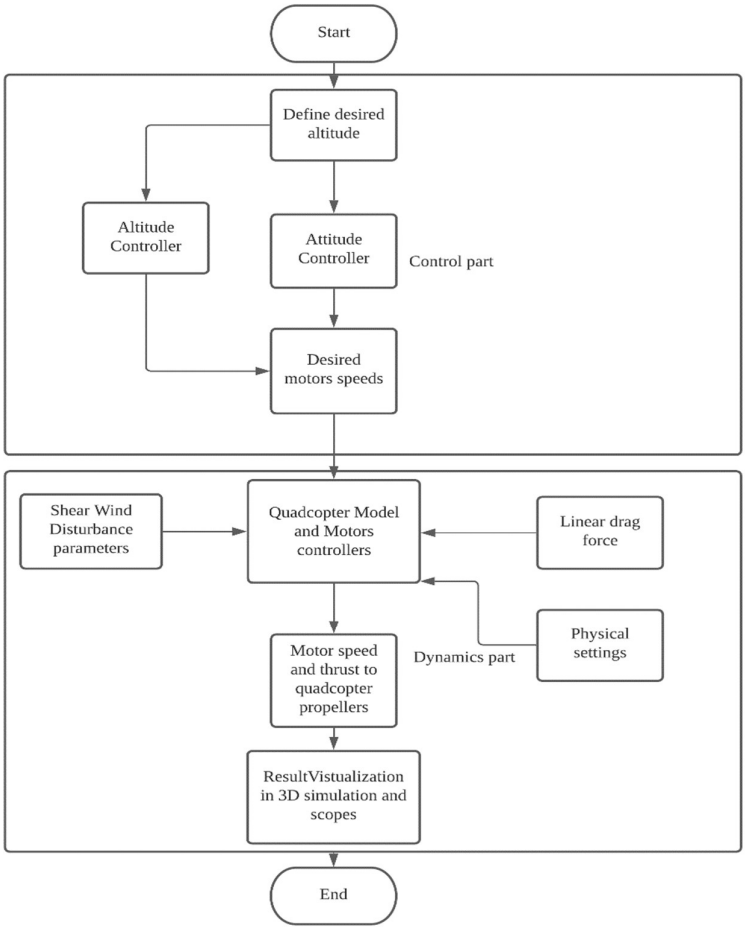
1.4 Aims and Objectives

- Master Quadcopter dynamic
- 3D simulation of Quadcopter altitude and attitude PID control with wind disturbance.

1.5 List of proposed deliverables for the project

- The quadcopter is simulated and controlled properly in MATLAB.
- Wind disturbance is correctly modeled.
- Control methodology and quadcopter dynamic are well elaborated.
- Related Work is briefly illustrated in form of the literature review.
- Result and analysis are provided.

1.6 Flow Chart



Literature Review

2.1 Quadcopter dynamics

Torque will be affected by the moments of inertia as [2]:

$$\tau_{x,y} = \begin{bmatrix} L(T_{\text{rotor 4}} - T_{\text{rotor 2}}) \\ L(T_{\text{rotor 3}} - T_{\text{rotor 1}}] \end{bmatrix} = \begin{bmatrix} \left(K_{\phi,p}(\phi_p{}' - \phi') + K_{\phi,D}(\phi_d{}' - \phi')\right) I_{xx} \\ \left(K_{\theta,p}(\theta_p{}' - \theta') + K_{\theta,D}(\theta_d{}' - \theta')\right) I_{yy} \end{bmatrix}$$

System equations of motion can be represented as[5]:

$$m\,{}^{B}\dot{\mathbf{V}} = -\ {}^{B}\boldsymbol{\nu} \times (m\,{}^{B}\mathbf{V}) + {}^{B}_{\mathcal{O}}\mathbf{R}\mathbf{g} + {}^{\mathcal{B}}\mathbf{T} - {}^{B}\mathbf{F}_D$$
$$\mathbf{I}\,{}^{B}\dot{\boldsymbol{\nu}} = -\ {}^{B}\boldsymbol{\nu} \times (\mathbf{I}\,{}^{B}\mathbf{V}) + \boldsymbol{\Gamma} + {}^{B}\boldsymbol{\tau} - {}^{B}\mathbf{M}_D$$
$${}^{\mathcal{O}}_{B}\mathbf{R} = \begin{bmatrix} C_\psi C_\theta & C_\psi S_\theta S_\phi - S_\psi C_\phi & C_\psi S_\theta C_\phi + S_\psi S_\phi \\ S_\psi C_\theta & S_\psi S_\theta S_\phi + C_\psi C_\phi & S_\psi S_\theta C_\phi - C_\psi S_\phi \\ -S_\theta & C_\theta S_\phi & C_\theta C_\phi + S_\psi S_\phi \end{bmatrix}$$

Where ${}^{B}\mathbf{F}_D$ ${}^{B}\mathbf{M}_D$ are drag force and moment, $\boldsymbol{\Gamma}$ is the gyroscopic moment, g is gravity, and I am the inertia.

Drag force and moment can be defined as[5]:

$${}^{B}\mathbf{F}_D = \text{sgn}(\ {}^{B}\mathbf{V}_{\text{app}}) \circ \mathbf{c}_f \circ {}^{B}\mathbf{V}_{\text{app}}\ {}^{\mathcal{B}}\mathbf{V}_{\text{app}}$$
$${}^{B}\mathbf{M}_D = \text{sgn}(\ {}^{B}\nu) \circ \mathbf{c}_m \circ {}^{B}\nu \circ {}^{B}\nu$$

Where $\circ$ is the Hadamard product, $\text{sgn}(\alpha)$ is signum function, $\mathbf{c}_f$ and $\mathbf{c}_m$ are translational and rotational drag constants vectors, ${}^{B}\mathbf{V}_{\text{app}}$ includes translational velocity ${}^{B}\nu$ and wind vector $w_{\mathbf{U}}$

$${}^{B}\mathbf{V}_{\text{app}} = {}^{\mathcal{B}}\mathbf{V} - {}^{B}_{\mathcal{O}}\mathbf{R}^{\mathcal{O}}_{W}\mathbf{R}^{W}\mathbf{U}$$
$$w_{\mathbf{U}} = [w_{U_u}\quad w_{U_v}\quad w_{U_w}]^{\top}$$

2.2 Fuzzy PID Quadcopter control

The Fuzzy-PID controller has more accuracies and is more capable of rejecting wind disturbances than the PID controller.

Equation of the PID controller [2]:

$$U(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{d}{dt} e(t)$$
$$e(t) = X_d(t) - X(t)$$

The Fuzzy PID control attempts to develop a continuous two-variable function between the PID parameters, absolute error value changes, and absolute error value[3].

Equation of the fuzzy PID controller [4]:

$$U_{Fuzzy-PID} = K_{p-Fuzzy}e(t) + K_{i-Fuzzy}\int_0^t e(\tau)d\tau + K_{d-Fuzzy}\frac{d}{dt}e(t)$$

2.3 Full quaternion-based Quadcopter attitude control

Another computationally efficient method for controlling quadcopter attitude is by describing the pose by a single quaternion $\boldsymbol{q}$ given by [13]

$$\boldsymbol{q} = \begin{bmatrix} \cos(\phi/2)\cos(\theta/2)\cos(\psi/2) + \sin(\phi/2)\sin(\theta/2)\sin(\psi/2) \\ \sin(\phi/2)\cos(\theta/2)\cos(\psi/2) - \cos(\phi/2)\sin(\theta/2)\sin(\psi/2) \\ \cos(\phi/2)\sin(\theta/2)\cos(\psi/2) + \sin(\phi/2)\cos(\theta/2)\sin(\psi/2) \\ \cos(\phi/2)\cos(\theta/2)\sin(\psi/2) - \sin(\phi/2)\sin(\theta/2)\cos(\psi/2) \end{bmatrix}$$

Where the roll angle ϕ is the rotation of the quadcopter around its X-axis is, the pitch angle θ is the rotation around it Y-axis is and yaw angle ψ is its rotation around the Z-axis.

The error in pose can be found by taking the Hamilton product $\boldsymbol{q}_{\text{err}} = \boldsymbol{q}_{\text{ref}} \otimes \boldsymbol{q}_m^*$

Where $\boldsymbol{q}_{\text{ref}}$ describes the required pose and $\boldsymbol{q}_m^*$ is the conjugate of the current pose A nonlinear P^2 controller with two loops can then be used for pose correction where the overall mathematical formulation is given as follows

$$\tau = -P_q \cdot \begin{bmatrix} q_1^{err} \\ q_2^{err} \\ q_3^{err} \end{bmatrix} - P_\omega \cdot \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}$$

Where P_q and P_ω are proportional gains. The controller will look similar to Fig. 1

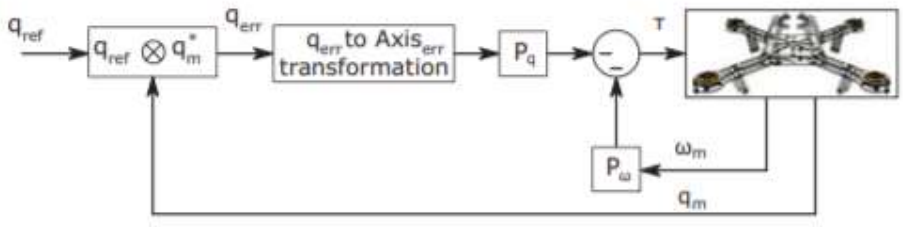


Figure 1. Non-linear P^2 controller

The advantage of this controller has a lower computational cost compared to using direction cosine matrices or Euler angle conversions in attitude controllers.

System Model

The overall system is split into two parts control and dynamics as shown below

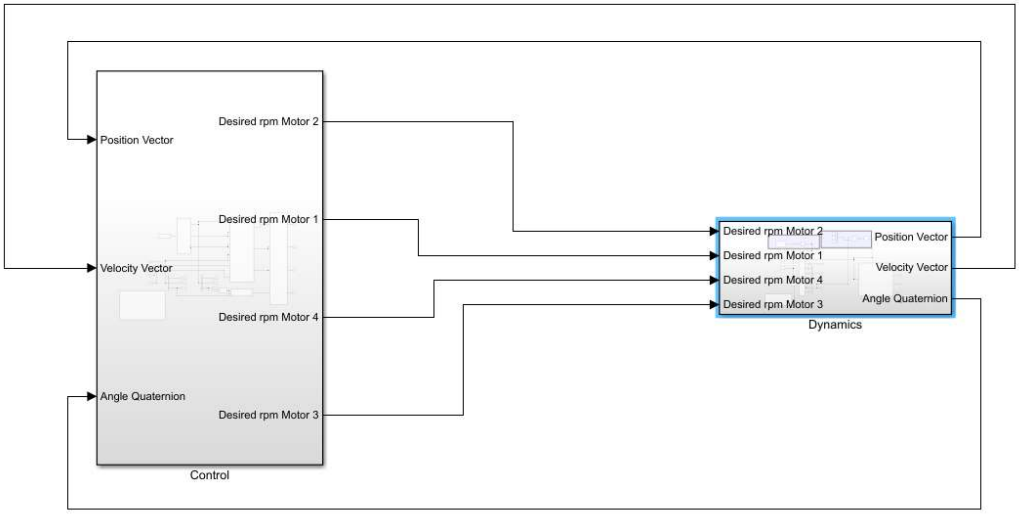


Figure 2. Overall system layout

The dynamics subsystem represents the quadcopter plant and has the desired RPM of each motor as the input. The output of the quadcopter plant is the position vector, velocity vector, or and the angle quaternion of the quadcopter body.

The following sections describe how the dynamics subsystem was modeled

3.1 Modelling the quadcopter

Simscape Multibody allows us to import ".STEP" files and connect them appropriately using rigid transformations to simulate the quadcopter. Simscape Multibody automatically calculates the inertial properties of the solids. Given below are the file names for the solid bodies and their properties.

File name	Mass (g)	Dimensions (mm)	Number of times used
plate_top.stp	31.7292	From file	2
Sensor suite	638	[60 60 15]	1
arm.stp	58.9152	From file	4
Motor base	0.00384845	Radius= 14 Length =5	4
bldc.stp	51	From file	4
Propeller.stp	24.9468	From file	4

Given below are the connections to create the quadcopter model

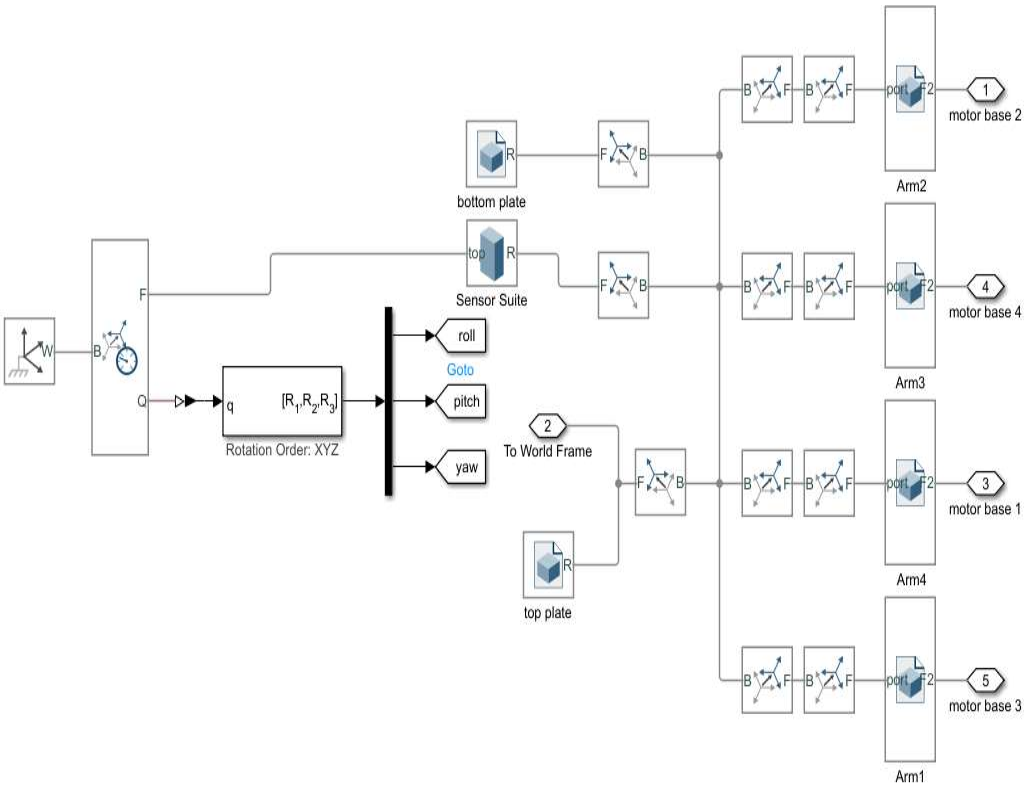


Figure 3. Main quadcopter chassis connections

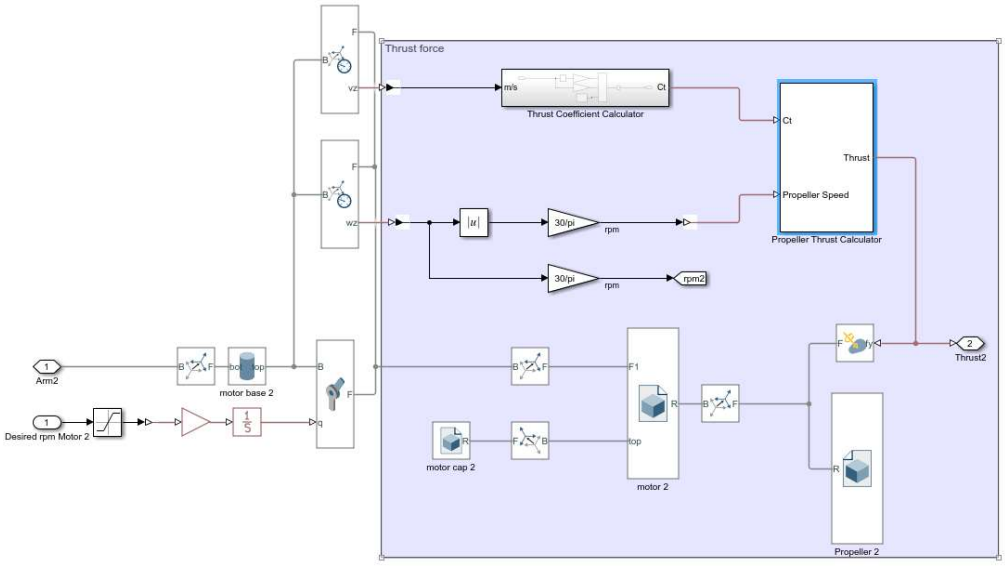


Figure 4. Connections for the motors and propellers

Finally, a ground of dimensions [1 1 0.1] meters is connected to the world frame as a reference during the simulation.

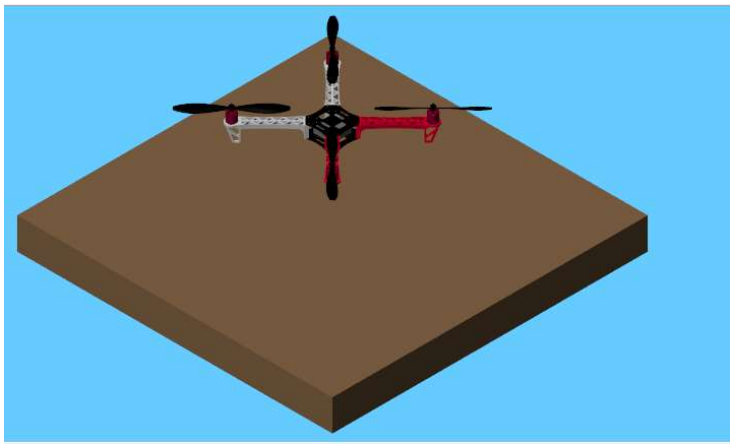


Figure 5. Quadcopter model

To simulate the motion of the quadcopter, a 6 degree of freedom joint is connected to the follower frame of the quadcopter chassis and a revolute joint is connected to the end of each arm. A transform sensor is connected to the 6-DOF joint to get the velocity, position, and angle quaternion which describes the pose of the quadcopter relative to the world frame. The revolute joint is connected to another transform sensor to obtain the velocity in the +z direction and the angular velocity of the joint.

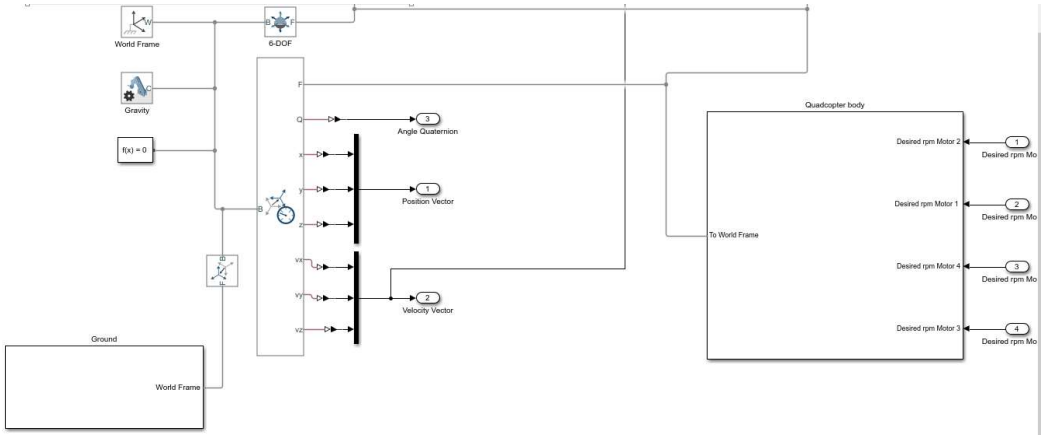


Figure 6. 6-DOF joint connection with the quadcopter

3.2 System Dynamics

Many forces and torques to many forces and torques act on the quadcopter. For simplicity, we consider only the torques are acting thrust force, roll-pitch-yaw torques, linear drag, and a wind force. The following subsections described how these forces were modeled in our system.

3.2.1 Thrust force

The thrust force is caused by the air getting pushed downwards by the propellers, which in turn creates a reaction force in the +Z direction of the propellers. Thrust force was added into our model as an external force in the +Z direction on each of the propellers. This thrust force is given by the equation

$$T = k_T \rho n^2 D^4$$

Where T is the thrust force, k_T is the thrust coefficient, ρ is the density of air, n is the propeller RPM and D is the diameter of the propeller.

The thrust coefficient k_T is a function of advance ratio J which is the distance advanced by the propeller in one revolution, Reynold's number(Re), and the Mach number of the tip(M_{tip}).

The advance ratio J is given by J = $\frac{u_0}{Dn}$

Where u_0 is the perpendicular speed of the propeller, in our case the velocity in the +z direction of the propeller.

The thrust coefficient k_T is thus a function of the blade configuration and parameters and can be approximated to the equation below for this propeller

$$K_t = -0.001589v_z^2 - 0.008716v_z + 0.1072$$

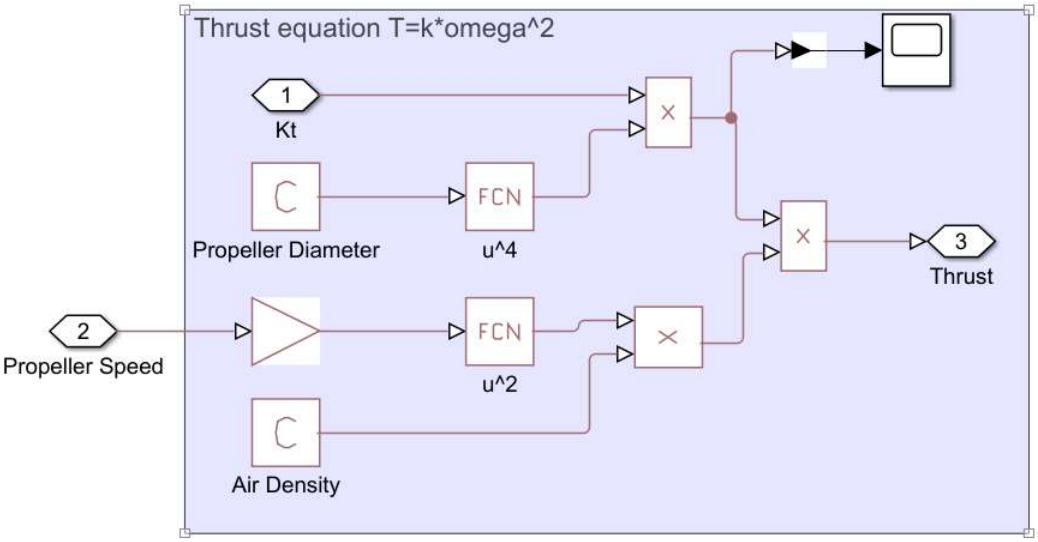


Figure 7. Thrust force calculator

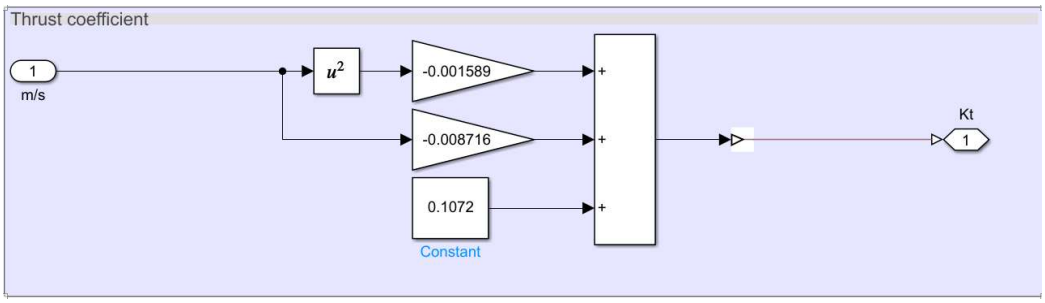


Figure 8. Thrust coefficient

Figure 7 and Figure 4 show how we have modeled thrust into our system. To add an external force in Simscape Multibody we can simply use an external force block where the input is the required force and output is connected to the required follower

frame. Calculation of this input force is shown in Figure 7. The input of the thrust force calculator is the propeller angular speed taken from the transformed sensor of the appropriate revolute joint.

3.2.2 Roll-Pitch-Yaw Torques

The attitude of a quadcopter is defined by its angle space $[\phi \ \theta \ \psi]$ where the roll angle ϕ is the rotation of the quadcopter about its X-axis, the pitch angle θ is the rotation about its Y-axis is and yaw angle ψ is its rotation about the Z-axis

The rotation of the propellers creates torques around the X, Y, and Z axes of the body frame of the quadcopter and is given by the equation

$$\begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \begin{bmatrix} lk(-\omega_2^2 + \omega_4^2) \\ lk(-\omega_1^2 + \omega_3^2) \\ \sum_{i=1}^{4} \tau_{M_i} \end{bmatrix}$$

$$\tau_{M_i} = b\omega_i^2 + I_M\dot{\omega}_i$$

Where τ is the appropriate torque, ω is the angular velocity of the rotors, l is the length of the arms and k is the thrust coefficient.

Because we applied an external thrust force on each propeller, we need not add the torques as external forces as Simscape Multibody automatically calculates and implements them.

3.2.3 Linear Air drag

To create a more realistic environment, we have added air resistance to the body of the quadcopter as it moves. The force on the quadcopter due to this air resistance can be represented as

$$F_{drag} = \begin{bmatrix} A_x & 0 & 0 \\ 0 & A_y & 0 \\ 0 & 0 & A_z \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix}$$

Where A_x, A_y, and A_z are the drag constants and $\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix}$ is the velocity vector of the quadcopter. In the frame of the quadcopter. A_x, A_y, and A_z must be taken negative.

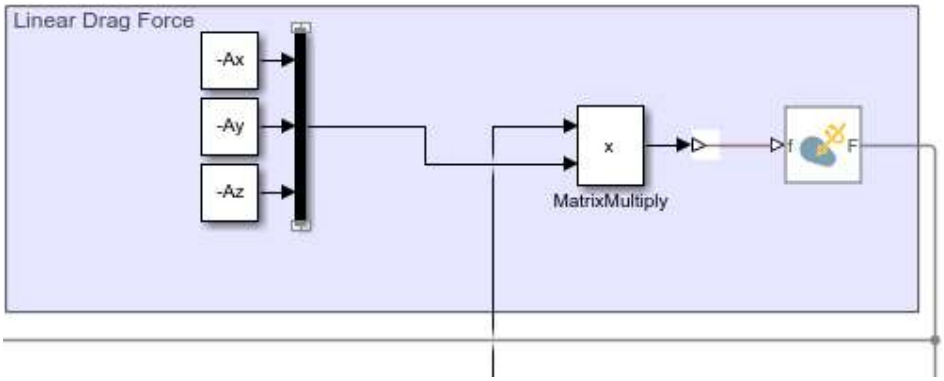


Figure 9. Drag force model

3.2.4 Wind disturbance force

We modeled the wind disturbance as pulses/gusts of wind using a pulse generator block. The wind disturbance force was generated using the Wind shear model and a Horizontal wind model provided by the aerospace toolbox in Simulink.

This block allows us to transform horizontal wind (north and east components) into body coordinates given wind speed, wind direction, and direction cosine matrix (DCM).

The output of the block is a velocity vector of the wind which we can transform into force given the density of air and area of effect. The pulse width, height of wind shear, angle of the wind, and wind speed can be changed by the user to change the characteristics of the wind.

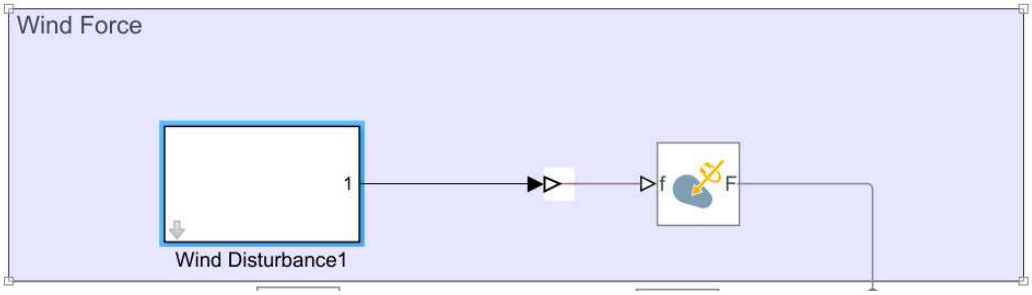


Figure 10. Wind disturbance model

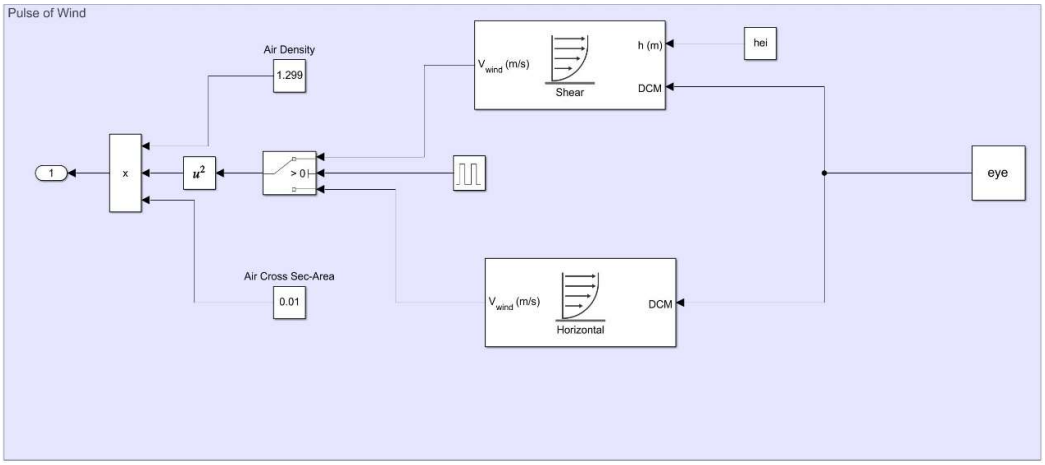


Figure 11. Wind pulse generator, the horizontal model wind velocity is zero

3.3 System Dynamics parameters

Given below is the table of system parameters

Constant name	Value	Units
D	0.254	m
ρ	1.225	kg/m^3
A_x	0.25	Kg/s
A_y	0.25	Kg/s
A_z	0.25	Kg/s

3.4 Control System

The control subsystem was then designed to control and obtain the required RPM for each of the motors on the quadcopter. The input for this subsystem is the position vector, velocity vector, and the Angle Quaternion. These inputs were obtained using a

transform sensor block connected to the 6-DOF joint in the dynamics subsystem.

Position and velocity vector can be written as

$$p = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \ and \ \ v = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{pmatrix}$$

Angle quaternions are an alternative to direction cosine matrices and Euler angles to describe the pose of the quadcopter. The quaternion $\boldsymbol{q}$ describing the pose of the quadcopter can be written as:

$$\boldsymbol{q} = \cos\left(\frac{\alpha}{2}\right) + \boldsymbol{u}\sin\left(\frac{\alpha}{2}\right)$$

$$\boldsymbol{q} = q_0 + q_1\boldsymbol{i} + q_2\boldsymbol{j} + q_3\boldsymbol{k}$$

Where u is the rotation axis (unit vector) and α is the rotation angle, and

$$\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1$$

The coming sections describe how these parameters are used to find the desired RPM of each motor which is then fed back into the dynamics block.

3.4.1 Altitude Generator

First, we must describe the trajectory of the quadcopter. This is done by the altitude generator block.

The altitude generator block takes home position i.e., the starting position of the quadcopter when it's at rest as the input and gives the final position, the desired velocity at the destination, and destination yaw angle as the outputs.

As this project is only concerned with altitude control when the quadcopter is hovering, we take the velocity at the destination as [0 0 0] and yaw angle as 0. The x and y position remains the same, z position is incremented by the user-defined altitude variable.

Figure 12. Altitude generator block

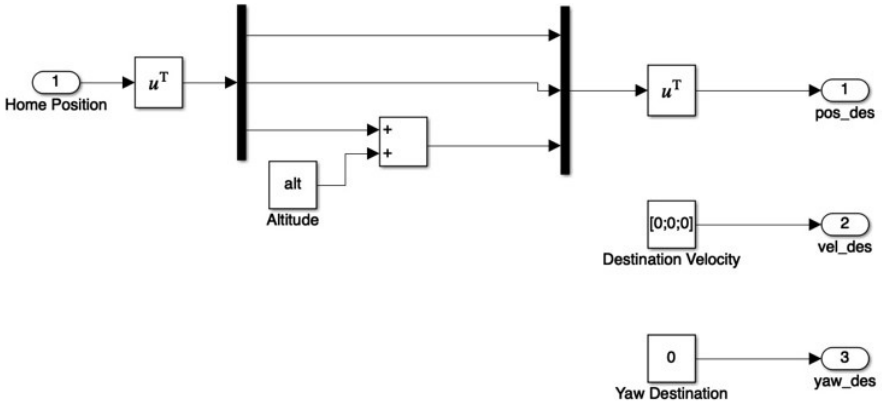


Figure 13. Schematics of Altitude generator block

3.4.2 Position Controller

Next, we must find the correction in the current roll pitch and yaw to stay at or reach the desired position. This is the function of the Position controller block. The desired velocity, desired position, and the desired yaw are taken as the input for the position controller block from the altitude generator. More inputs namely the current position, current velocity, and the angle quaternion are also taken from the world frame.

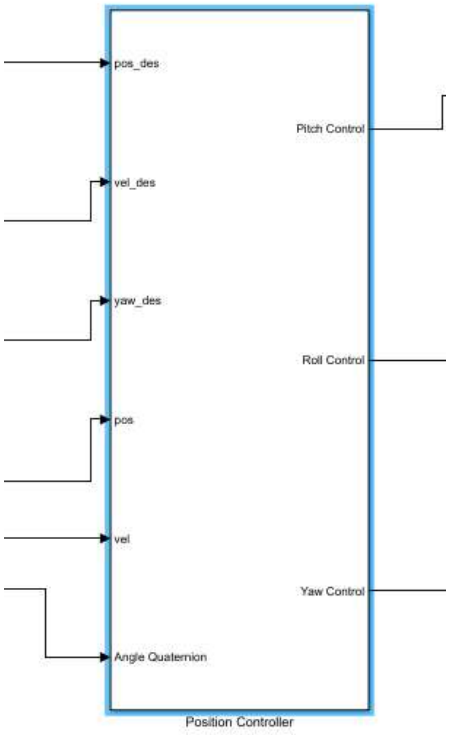


Figure 14 Position controller block

Inside the Position controller block, a mask named Roll Pitch and Yaw Controller takes these inputs and gives the destination (desired) Yaw, Roll, and Pitch. To do this we first find the control vector V

$$V = [(V_{des} - V_{cur}) + \ K_p * (P_{des} - P_{cur}) \ ]$$

Where K_p is a proportional gain for the Position vector.

The control vector must be rotated to the world frame to describe the pose required To rotate the control vector we take its Hamilton product with the current angle quaternion Q as shown

$$^{w}V = Q^{-1B}VQ$$

Where ^{w}V the control vector in the world frame is, Q is the angle quaternion of the quadcopter which describes its current pose, and ^{B}V describes the control vector in the body frame.

Quaternion operations can be easily done using the Aerospace Toolbox provided by MATLAB. A gain Kd is applied to the control vector. The control vector now

dimensionally represents an acceleration arc in the desired direction, dividing appropriately by g =9.81 gives the angle subtended by this arc which are the required pitch yaw and roll angles. It is worth noting that we are not dividing two vectors directly, roll and pitch subtend their separate acceleration arcs towards the desired position which must be divided by g to find the destination (desired) angle.

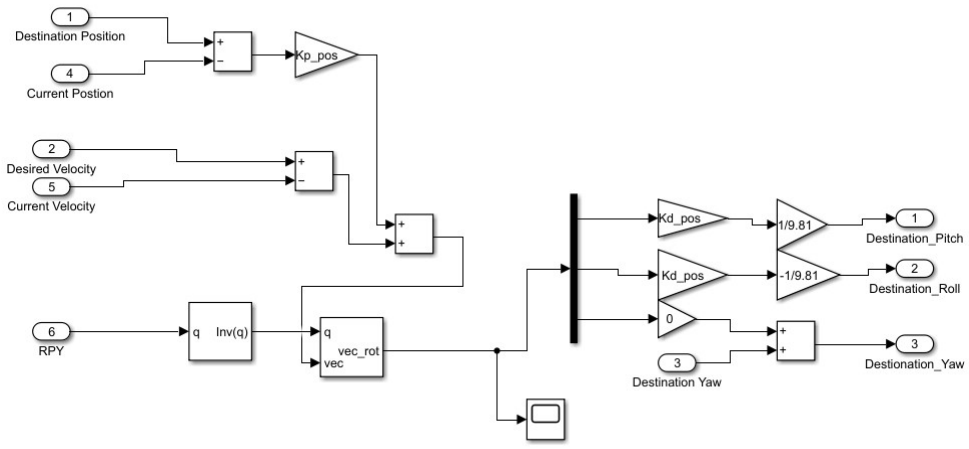


Figure 15. Inside the Roll Pitch Yaw controller

As we don't wish for the Quadrotor to rotate about its Z-axis, we keep the desired Yaw to be equal to zero. The output from the Roll Pitch Yaw controller is the desired Roll, Pitch, and Yaw. It is then subtracted from the current Roll, Pitch, and Yaw and then passed through to a PID controller with a Proportional constant equal to 8.5 and an Integral constant equal to 5, and the Derivative constant equal to 40. A saturation of 60^0 to -60^0 is applied to the destination pose angles so that the quadcopter does not destabilize.

The following Simulink model shows what was described above; the corrections for pitch roll and yaw are passed onto the motor mixer block.

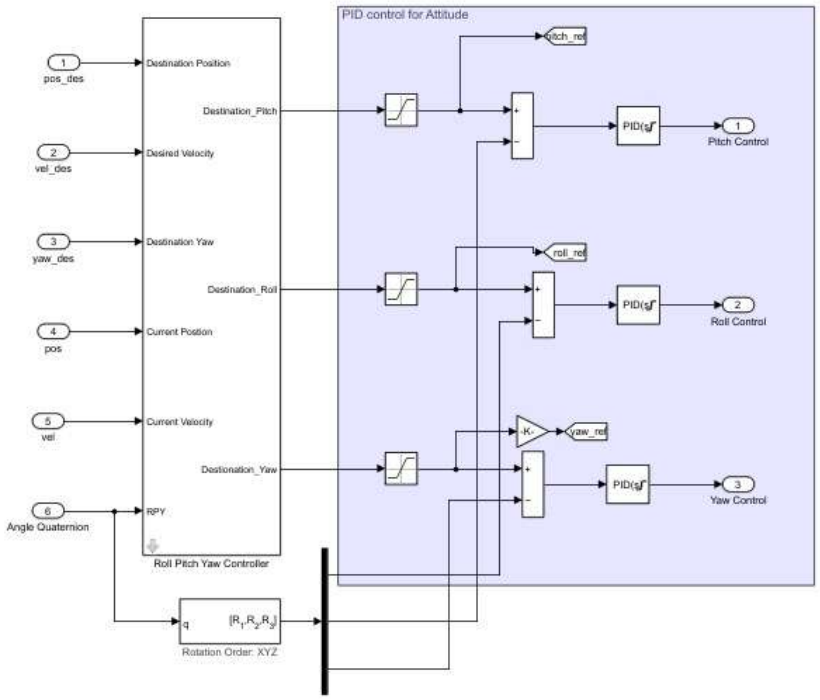


Figure 16. Inside the position controller block; PID control for attitude

PID parameters	Values	Control
Proportional (K_p)	8.5	Attitude
Derivative (K_d)	40	Attitude
Integral (K_i)	5	Attitude
Proportional (K_p pos)	0.0175	Position
Derivative (K_d pos)	0.85	Position

3.4.3 Altitude Controller

The altitude controller block calculates the RPM required to maintain or reach the desired altitude. The input is provided to the altitude controller block is the required altitude that needs to be attained (desired altitude) and the current altitude of the quadcopter.

Inside the altitude controller subsystem, the required altitude is first subtracted with the current altitude and then passed through a PID controller. The correction

altitude must be then converted to an RPM correction, to do this it is passed through a gain of $2\pi/(0.254^2 * \sqrt{1.225})$ obtained from the equation given below

$$C_t = \frac{T}{\rho * \ n^2 * \ D^4}$$

Here, C_t is the thrust coefficient and we assume that the thrust and Ct are unity for simplicity.

The correction RPM must then be added by a constant so that when it reaches the required altitude the motors just don't stop spinning and the quadcopter keeps hovering. This minimum hover speed was calculated to be 700 RPM. The resultant is the output and is called throttle control; this throttle RPM is equally fed into all the motors.

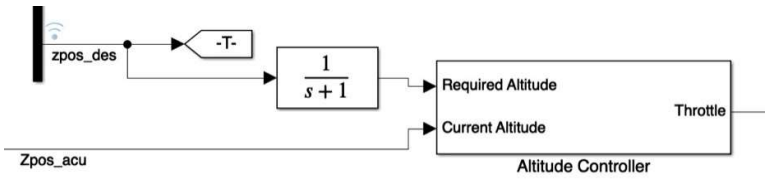


Figure 17. Altitude controller block

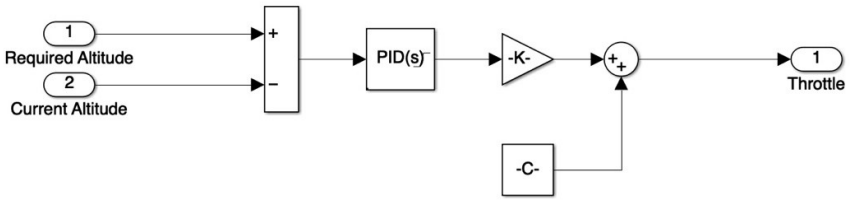


Figure 18. Altitude controller system

PID parameters	Values
Proportional (K_p)	0.15
Derivative (K_d)	0.475
Integral (K_i)	0.0275

3.4.4 Motor Output

The final step in the controller subsystem is to distribute the motor RPM's this is done by the motor mixer block. Outputs from the position controller and the altitude controller are taken as input and the output is the desired RPM for each of the four motors of the quadcopter. The equations below describe the motor RPM distribution to maintain altitude and perform the required pose correction to maintain position.

$$RPM_{moto} = Throttle\ \ control - Pitch\ \ control + Roll\ \ Control - Yaw\ \ Control$$

$$RPM_{motor2} = Throttle\ \ control - Pitch\ \ control - Roll\ \ Control + Yaw\ \ Control$$

$$RPM_{mot} = Throttle\ \ control + Pitch\ \ control + Roll\ \ Control + Yaw\ \ Control$$

$$RPM_{moto} = Throttle\ \ control + Pitch\ \ control - Roll\ \ Control - Yaw\ \ Control$$

The figure below shows the Simulink model which achieves this purpose.

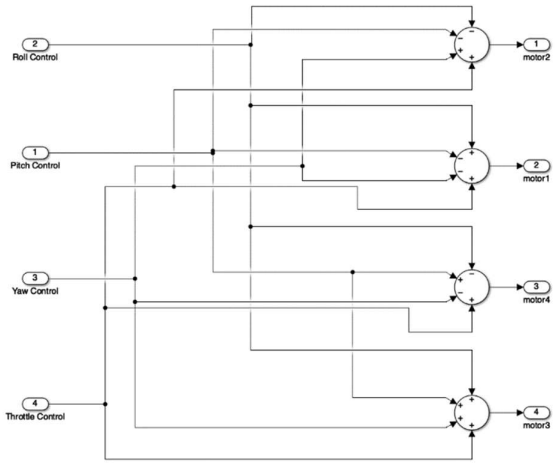


Figure 19. Motor mixer block

3.5 Feedback

The motor RPMs are fed into the revolute joint to simulate propeller rotation. To do this we first pass the RPM signal through saturation of 10000 RPM and a gain block

of $\frac{\pm 2\pi}{60} rad/sec$ to convert the RPM to the angular velocity. In our quadcopter model, alternate propellers rotate in opposing directions i.e. propellers on one diagonal arm rotate clockwise and the other diagonal arm rotate anticlockwise.

The input of the revolute joint is a physical signal containing the desired trajectory of the follower frame relative to the base frame of the revolute joint. We can obtain this by using the PS-Integrator block. The figure below describes how the desired RPM is fed into each revolute joint.

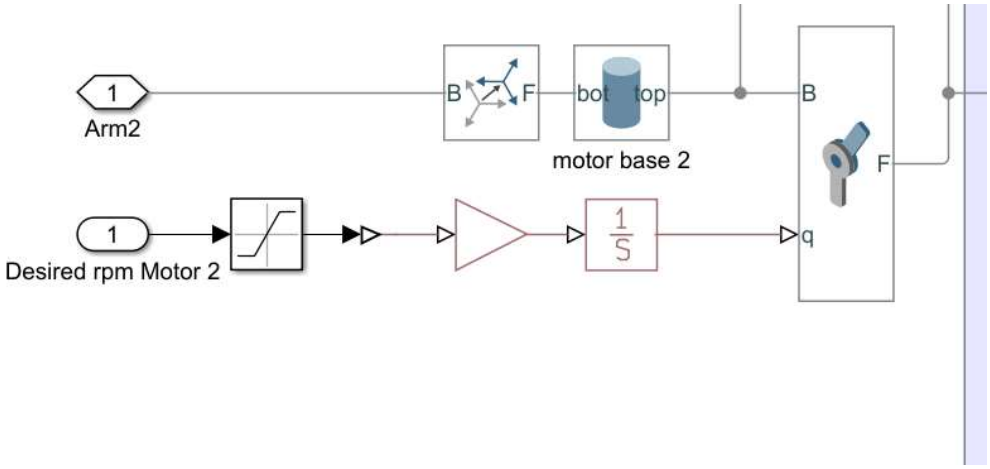


Figure 20. Feedback into revolute joint

Result and Discussion

4.1 Quadcopter without PID attitude control under normal wind disturbance

The result is simulated under normal wind speed conditions which are 11mph. Parameters that are used to run the result is shown as flowing:

Shear Wind disturbance speed	Wind direction	Wind Pulse	Stop time
11 mph or 5 meter/second	From southwest to northeast	50%	50 seconds

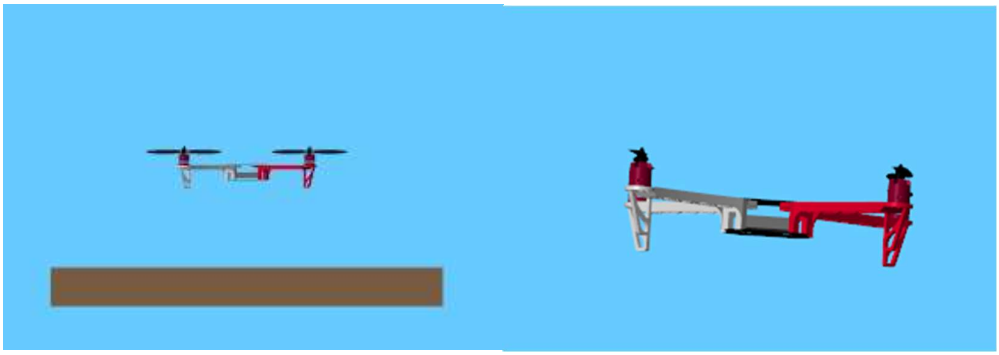


Figure 21. No control attitude frame 1 and frame 2

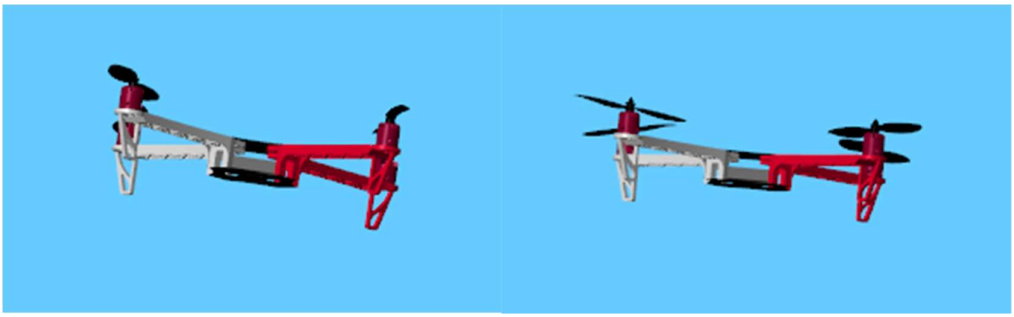


Figure 22. No control attitude frame 3 and frame 4

Figure 23. No control global frame 1

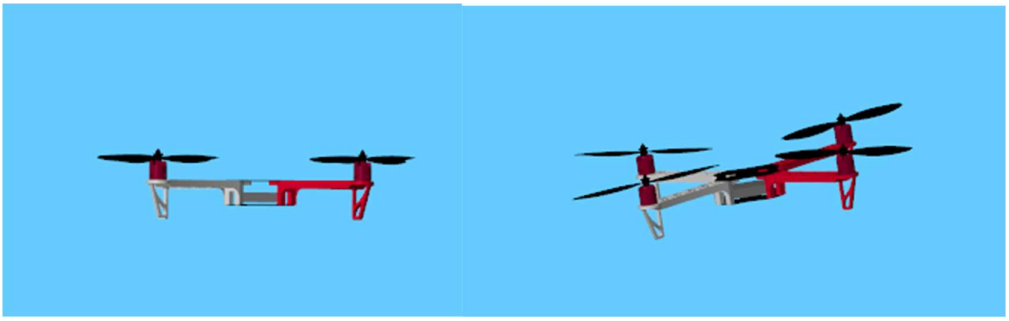


Figure 24. No control attitude frame 5 and frame 6

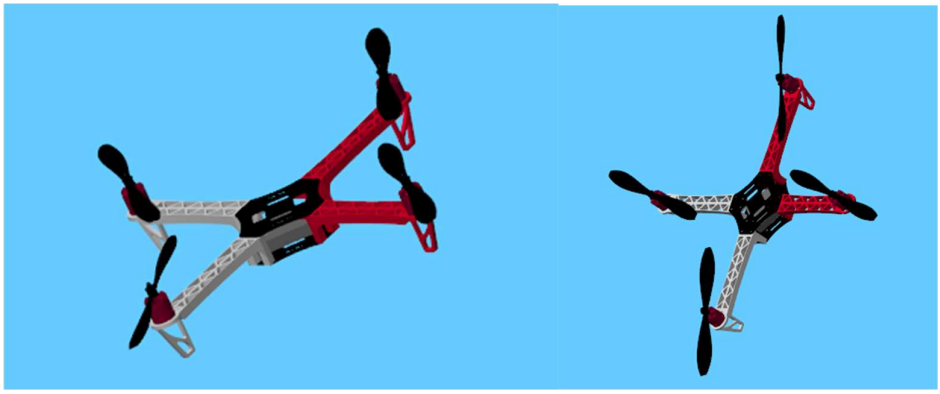


Figure 25. No control attitude frame 7 and frame 8

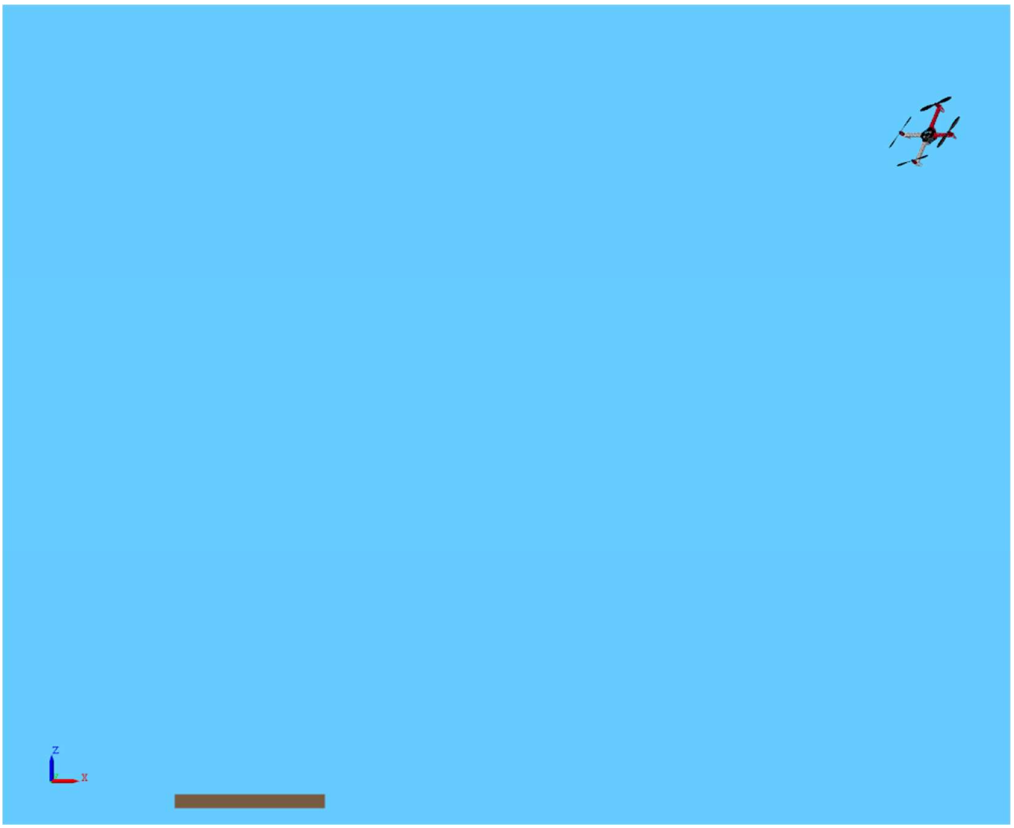


Figure 26. No control global frame 2

The no attitude control frames show that the quadcopter flies in an unstable manner due to the impact of southwest shear wind. The pose of the drone cannot balance.

4.2 Quadcopter with PID attitude control under normal wind disturbance

The result is simulated under normal wind speed conditions which are 11mph. Parameters that are used to run the results shown as flowing, it is the same as without a control model.

Shear Wind disturbance speed	Wind direction	Wind Pulse	Stop time
11 mph or 5 meter/second	From southwest to northeast	50%	50 seconds

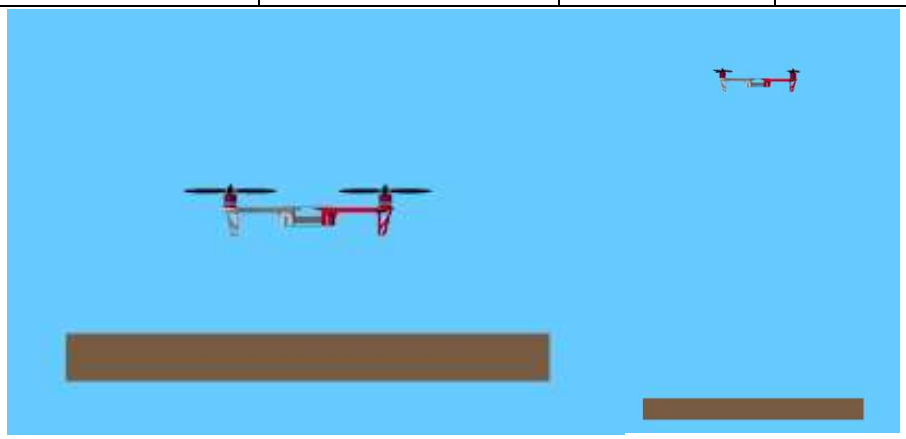


Figure 27. PID control attitude frame 1 and frame 2

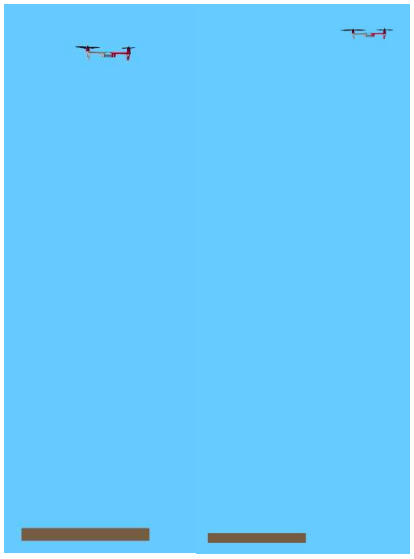


Figure 28. PID control attitude frame 3 and frame 4

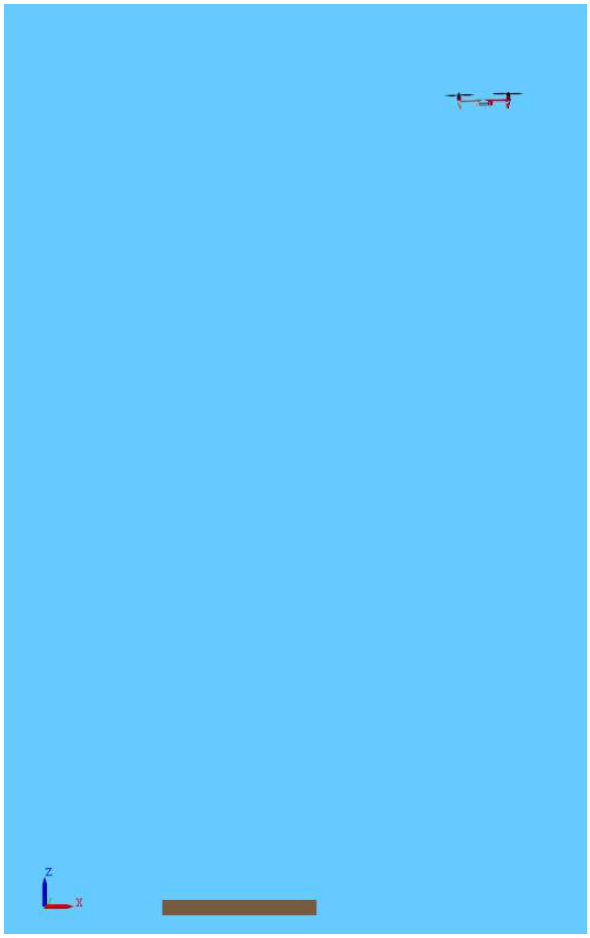


Figure 29. PID control global frame 1

We can observe from the frames that when PID control is applied, the quadcopter performance is much better than the no control model. Quadcopter flies in a very stable manner and attitude is perfectly maintained due to PID controller implementation and well PID parameters.

4.3 Quadcopter with PID attitude control under snowstorm wind disturbance

The result is simulated under severe wind speed conditions on which is 35mph (wind speed for a snowstorm). Parameters that are used to run the result is shown as flowing:

Shear Wind disturbance speed	Wind direction	Wind Pulse	Stop Time
35 mph or 15 meter/second	From southwest to northeast	50%	50 seconds

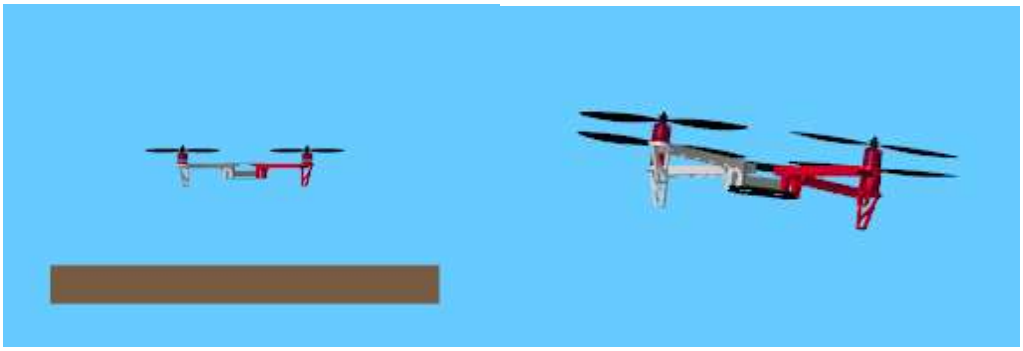


Figure 30. No control attitude frame 1 and frame 2

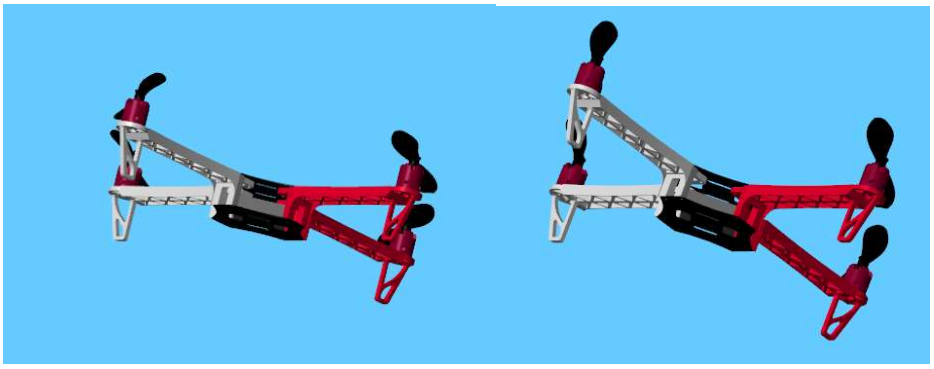


Figure 31. No control attitude frame 3 and frame 4

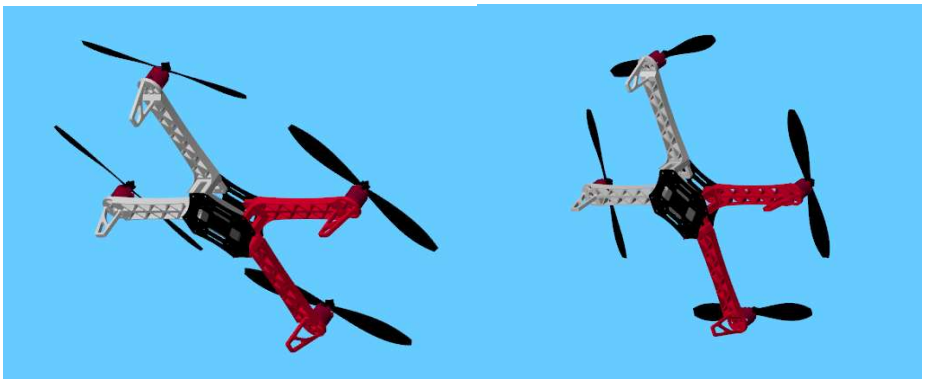


Figure 32. No control attitude frame 5 and frame 6

Figure 33. No control global frame 1

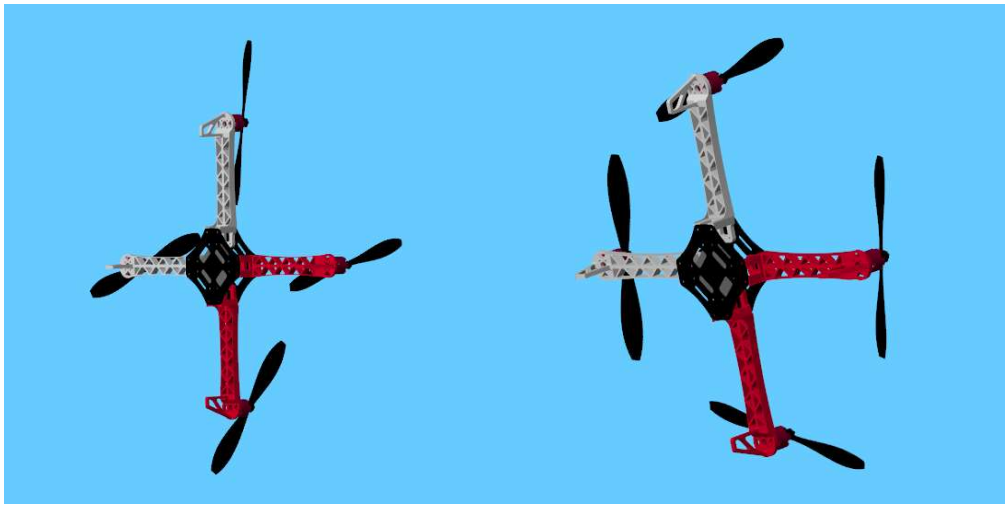


Figure 34. No control attitude frame 5 and frame 6

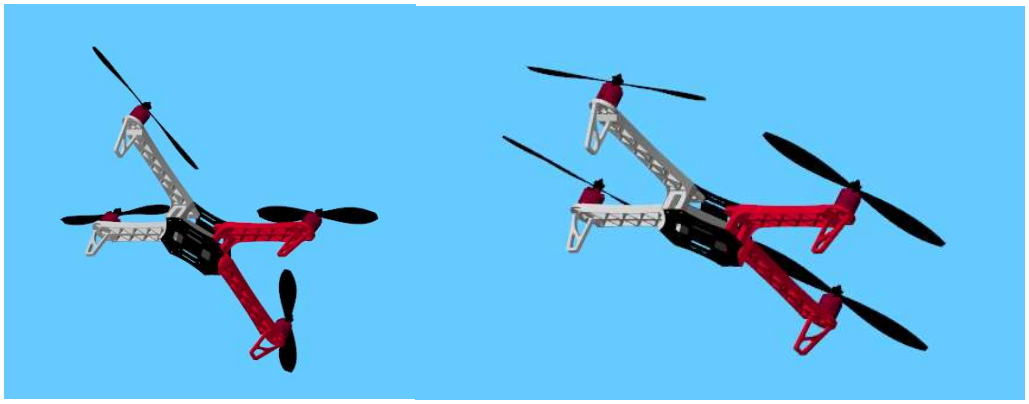


Figure 35. No control attitude frame 5 and frame 6

Figure 36. No control global frame 2

It can be observed that due to high wind disturbance, the ON-OFF control model easily loses its attitude and flies away under a short time duration.

4.4 Quadcopter with PID attitude control under snowstorm wind disturbance

The result is simulated under severe wind speed conditions which are 35mph (wind speed for a snowstorm). Parameters that are used to run the result are shown as flowing, it is the same as without a control model.

Shear Wind disturbance speed	Wind direction	Wind Pulse	Stop time
35 mph or 15 meter/second	From southwest to northeast	50%	50 seconds

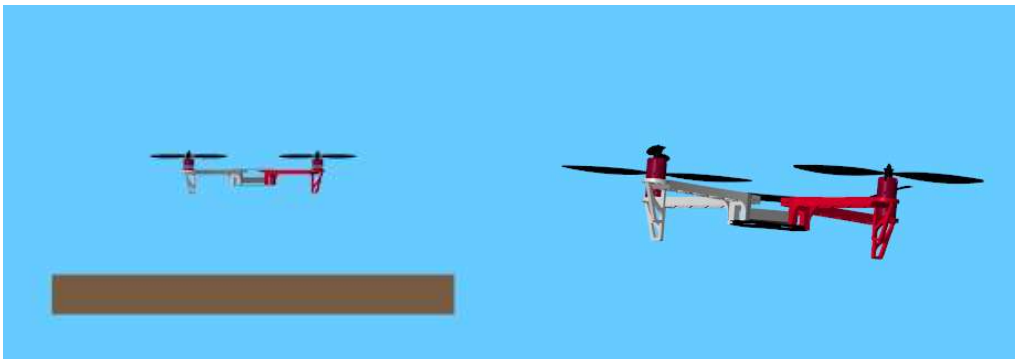


Figure 37. PID control attitude frame 1 and frame 2

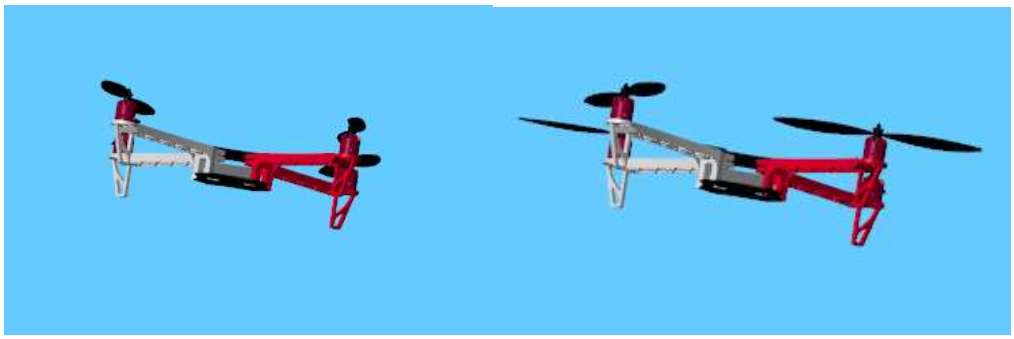


Figure 38. PID control attitude frame 3 and frame 4

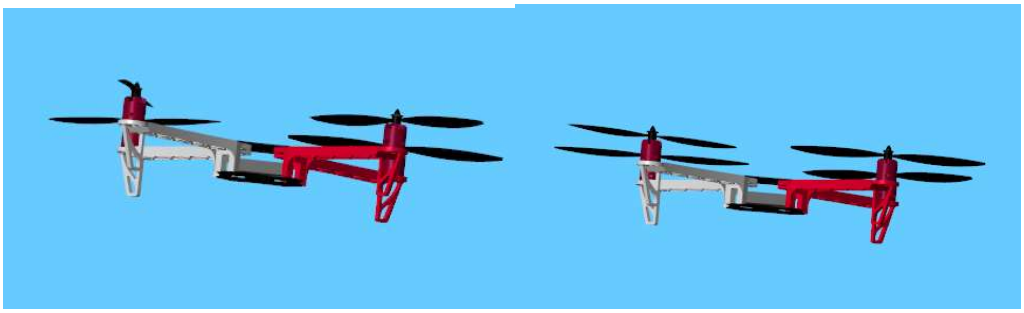


Figure 39. PID control attitude frame 5 and frame 6

Figure 40. PID control global frame 1

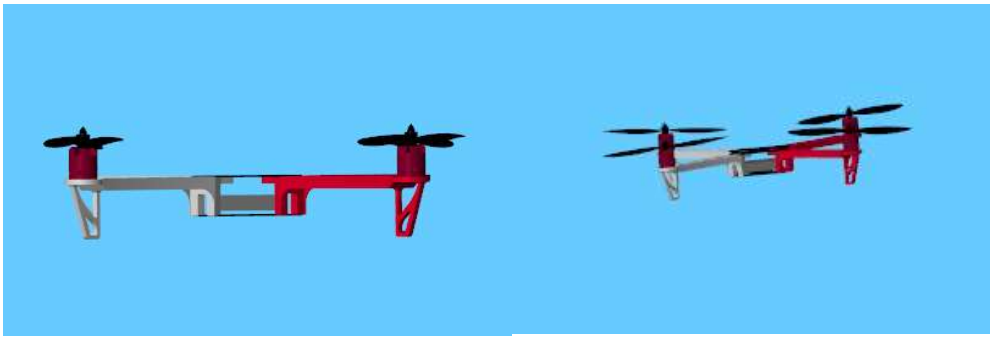


Figure 41. PID control attitude frame 7 and frame 8

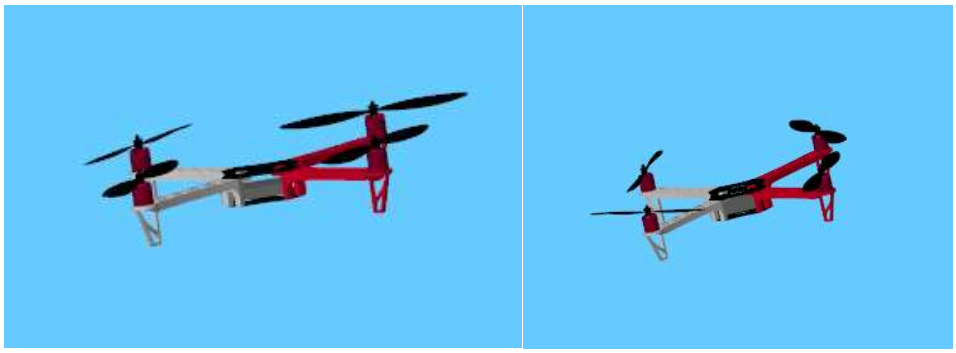


Figure 42. PID control attitude frame 9 and frame 10

Figure 43. PID control global frame 2

It can observe that due to high wind disturbance, the altitude is also fluctuating but the overall desired altitude and attitude are maintained within an acceptable range. It is very impressive that the quadcopter can survive and work in such hazard environment.

4.5 Scopes, analysis, and discussion

There were four conditions simulated while testing the altitude control on the Quadcopter. First was the condition where no wind was applied just to check whether the altitude controller was working properly. Second, a small wind disturbance was applied of 0.7m/s from the southwest direction. Third, a normal wind disturbance of 5m/s velocity was applied also from the southwest direction and lastly 15m/s which represent snowstorm wind disturbance. The results were compared and discussed in the following subsections. For each wind disturbance, three types of controller were also tested to check the performance which was, No Controller, Simple Feedback Controller (Bang-Bang Controller), and a PID Controller with the parameters stated above in the report.

4.5.1 No Altitude Control in no wind situation

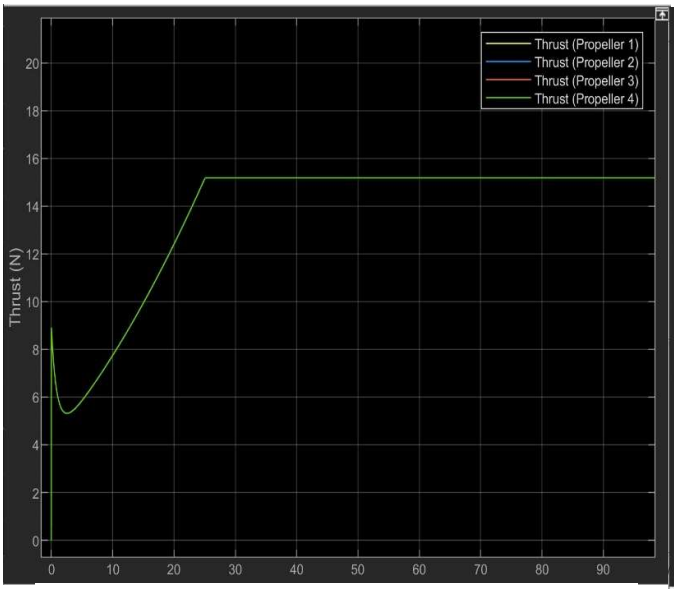


Figure 44. Propeller Thrust No wind No control

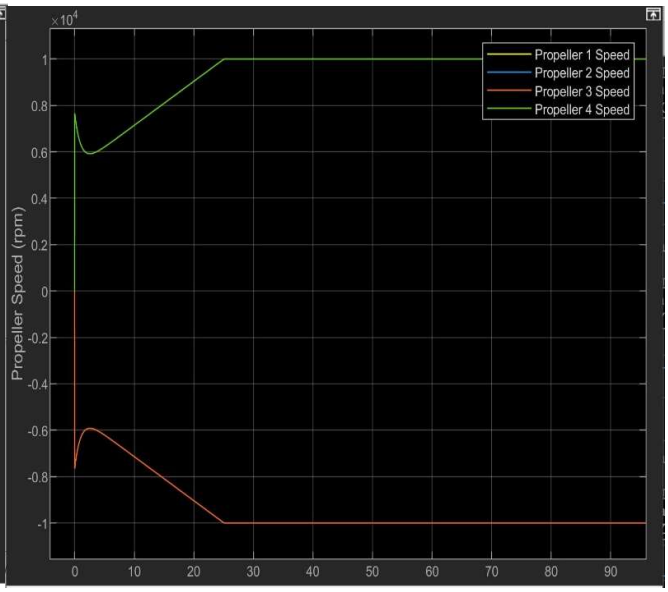


Figure 45. Propeller Speed No Wind No control

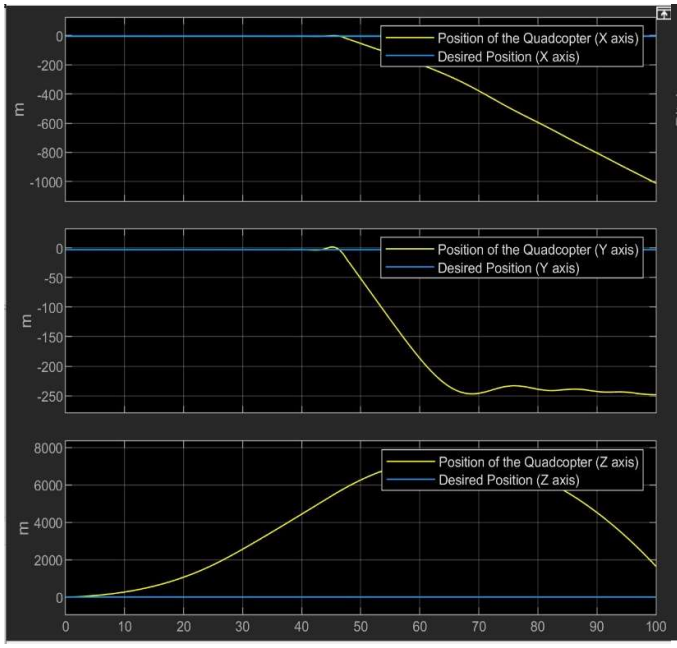


Figure 46. Position No Wind No control

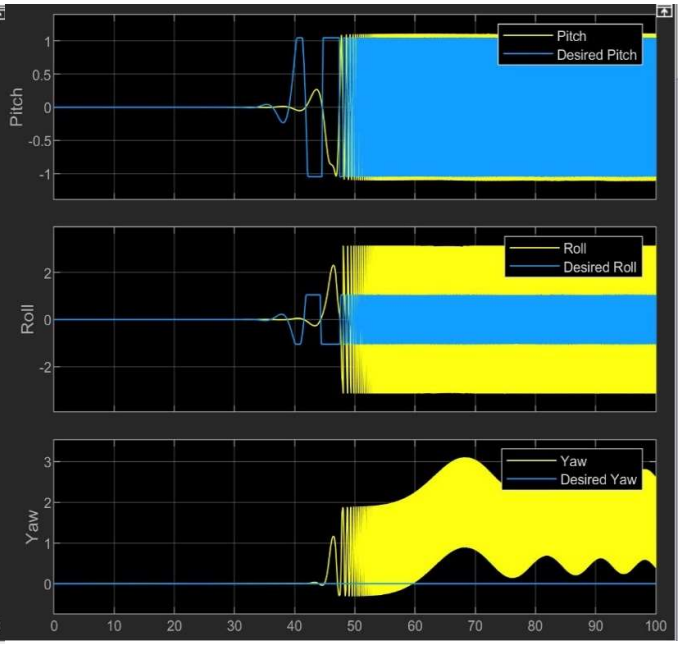


Figure 47. Attitude No Wind No Control

In this case, no altitude control is implemented so in this model the quadcopter once started will never attain the desired position and continues to move in a positive Z-axis direction as there is no wind disturbance. The position, altitude, propeller thrust, and propeller speeds areas provided

4.5.2 Simple Feedback Control in no wind situation

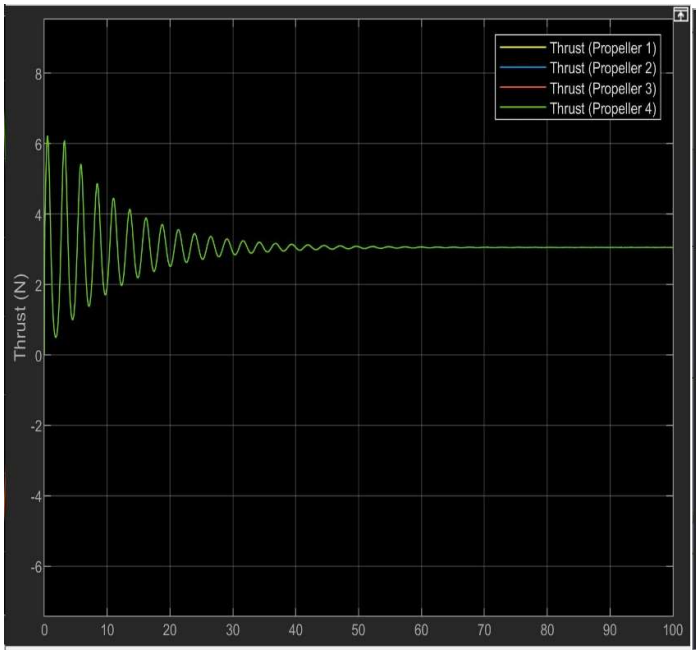


Figure 48. Propeller Thrust No Wind Simple Feedback

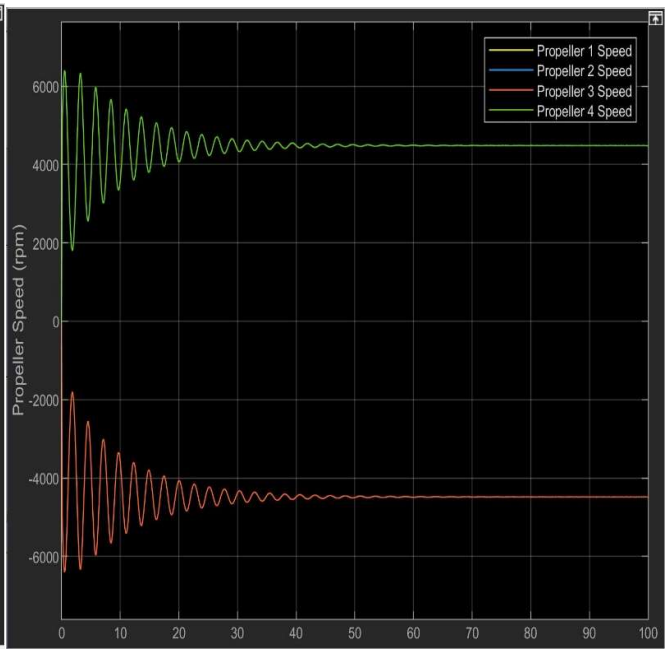


Figure 49. Propeller Speed No Wind Simple Feedback

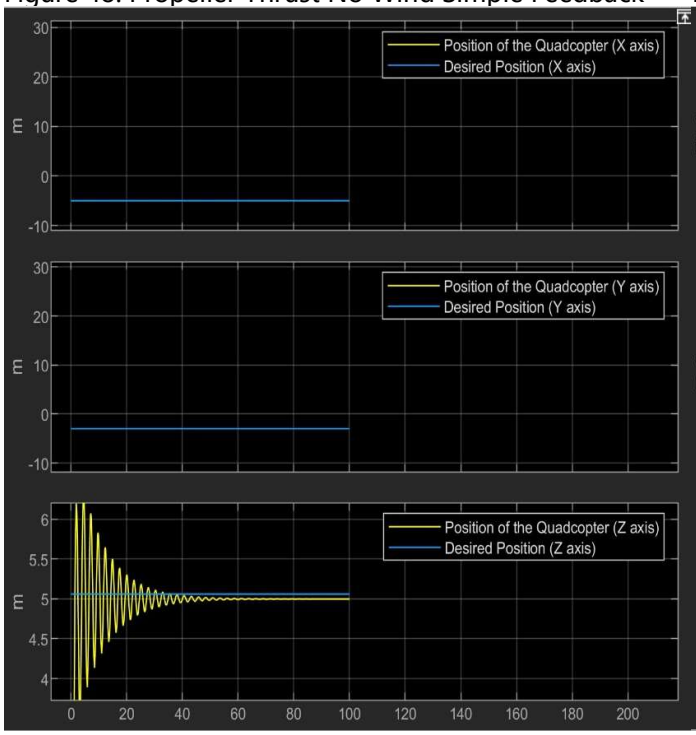


Figure 50. Position No Wind Simple Feedback

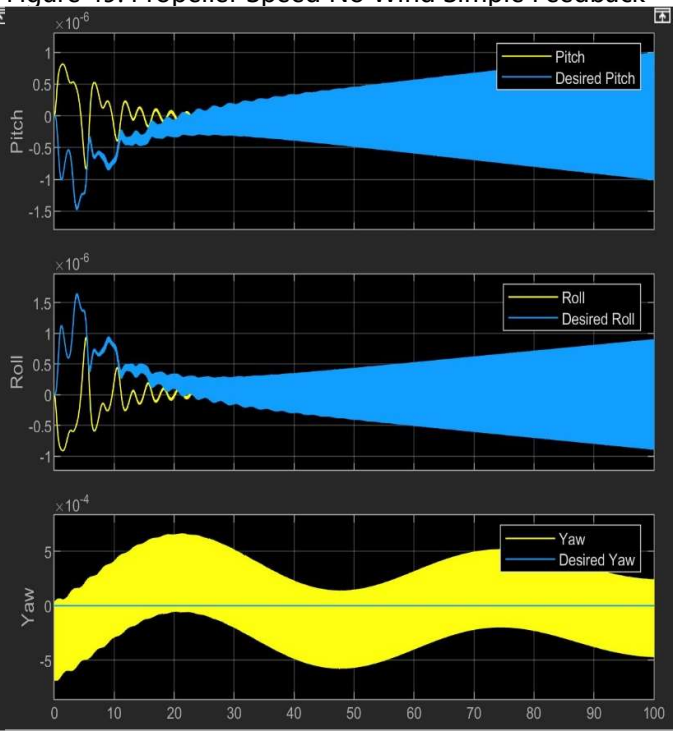


Figure 51. Attitude No Wind Simple Feedback

This case is similar to the previous case but a proportional block is used in Simulink to implement a simple feedback control. This type of ON-OFF control is also

known as Bang Bang Controller. In this mode when the quadrotor attains its desired position on the Z-axis the propellers are turned off which makes the quadcopter deviate from desired Z-axis so, to achieve the desired position the propeller is again turned on, this results in the quadcopter oscillating about the desired altitude along the Z-axis. The position, altitude, propeller thrust, and propeller speeds areas are provided.

4.5.3 PID Control in no wind situation

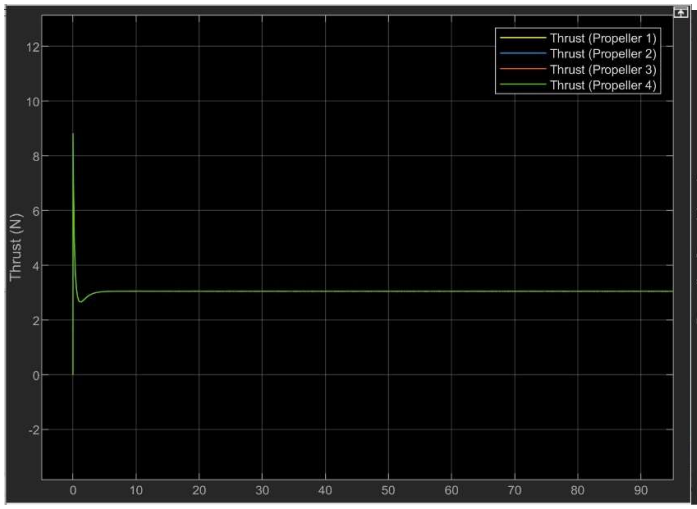


Figure 52. Propeller Thrust No Wind PID Controller

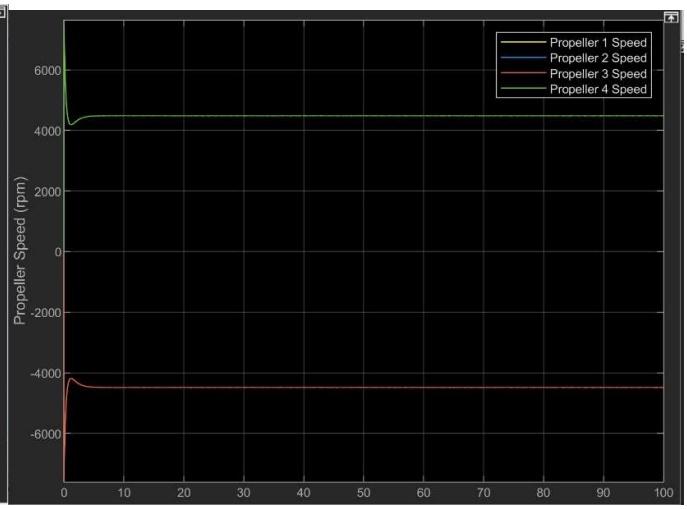


Figure 53. Propeller Speed No Wind PID Controller

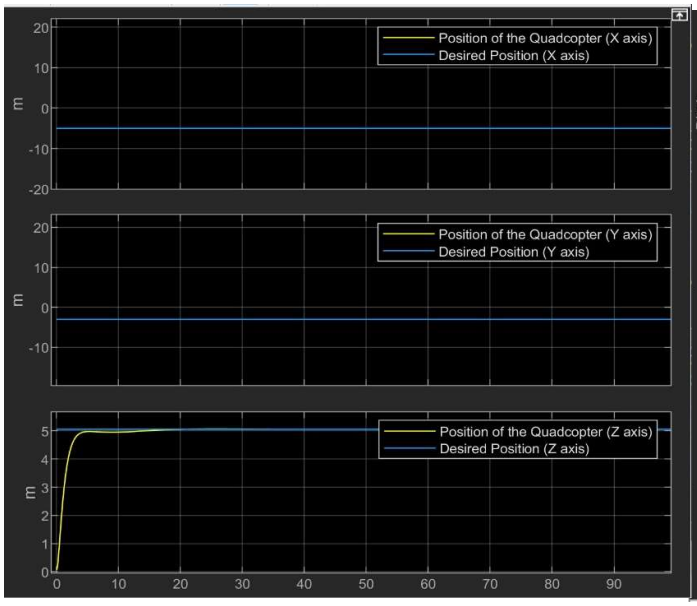


Figure 54. Position No Wind PID Controller

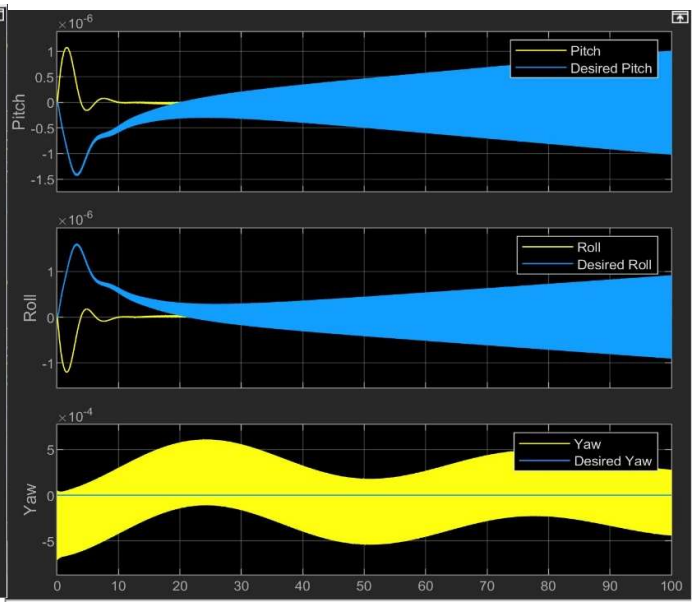


Figure 55. Attitude No Wind PID Controller

In this case, a PID controller is used to maintain the desired altitude. By using the PID controller the quadcopter can maintain the desired altitude without any deviations. The scopes are as provided

4.5.4 No Control in small wind situation (0.7m/s)

A small Wind disturbance of 0.7m/s (1.565 Miles per hour) is added to the model by using wind shear model blocks available in the Aerospace Toolbox. To make wind disturbances close to the real-world scenarios a pulse generator and switch case blocks are used. By using these blocks the wind disturbance is present for 5 seconds and no wind for the next 5 seconds.

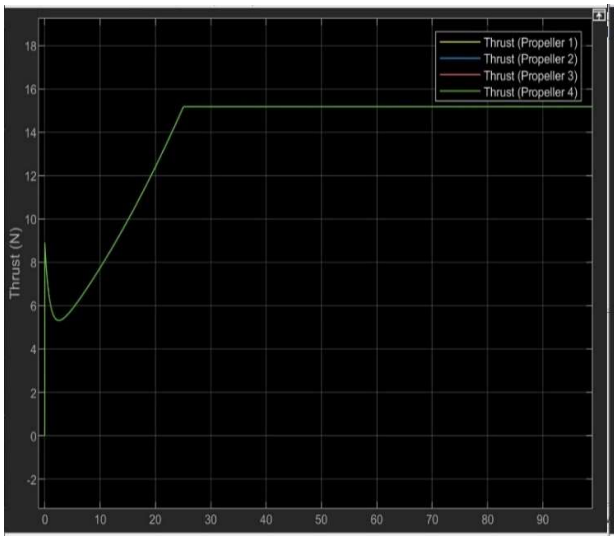


Figure 56. Propeller Thrust Small Wind No Control

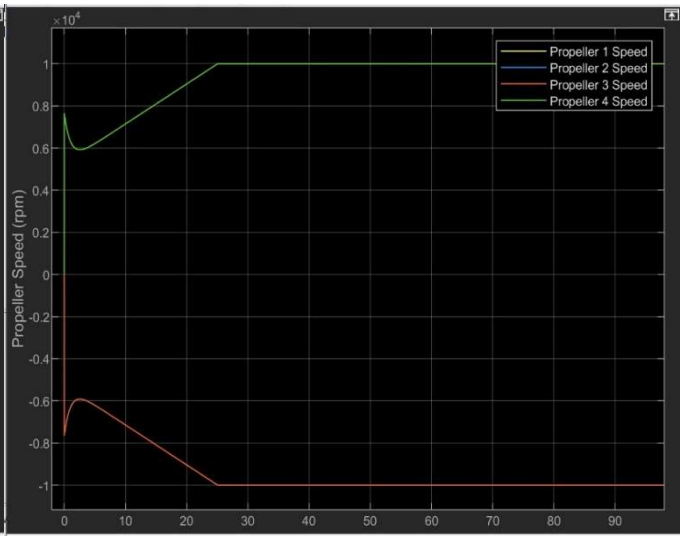


Figure 57. Propeller Speed Small Wind No Control

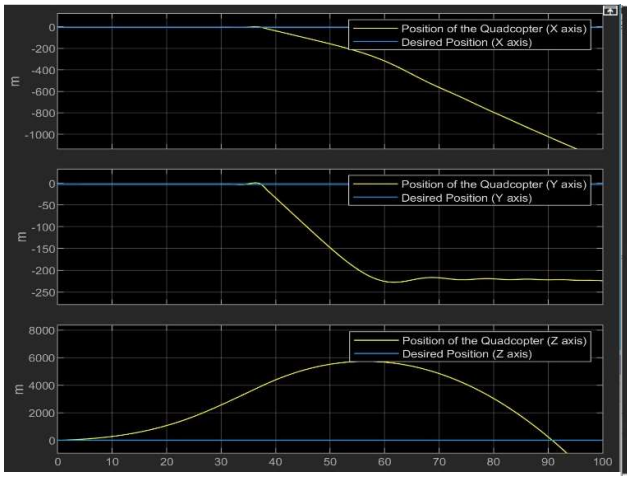


Figure 58. Position Small Wind No Control

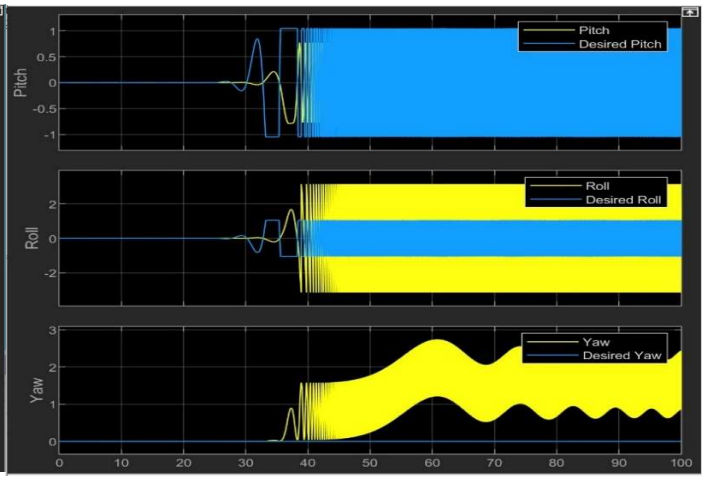


Figure 59. Attitude Small Wind No Control

When the wind conditions are given for no control model the quadrotor is unable to achieve the desired altitude and flies away from the desired reference frame.

4.5.5 Simple Feedback in Small Wind Disturbance Situation

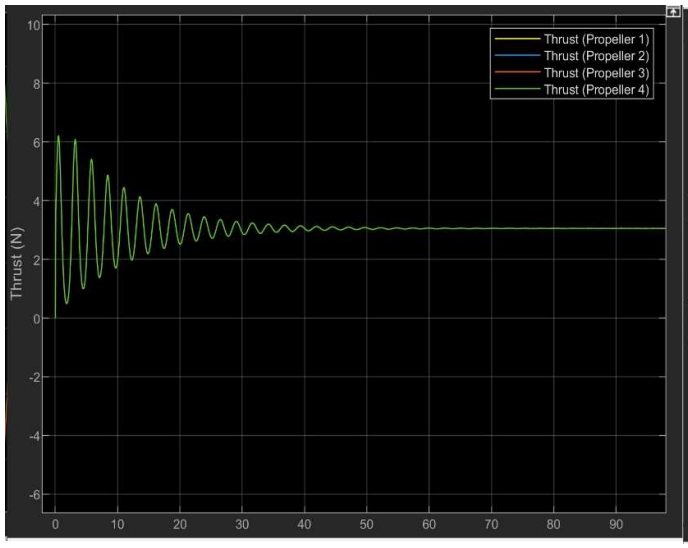


Figure 60 Propeller Thrust Small Wind Simple Feedback Control

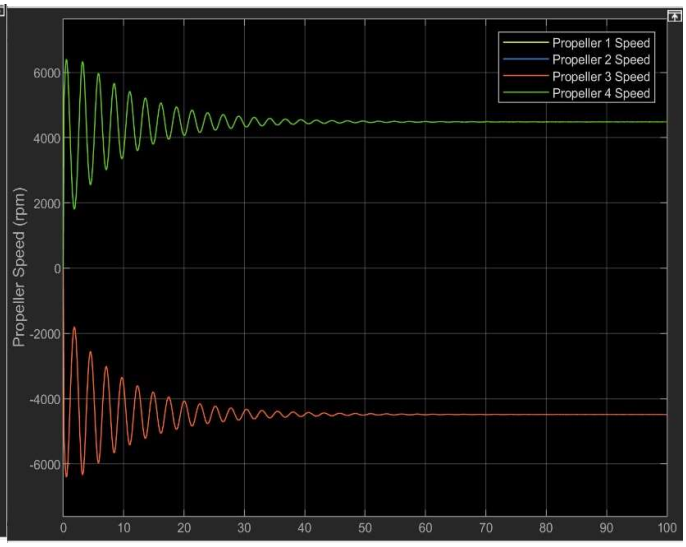


Figure 61 Propeller Speed Small Wind Simple Feedback

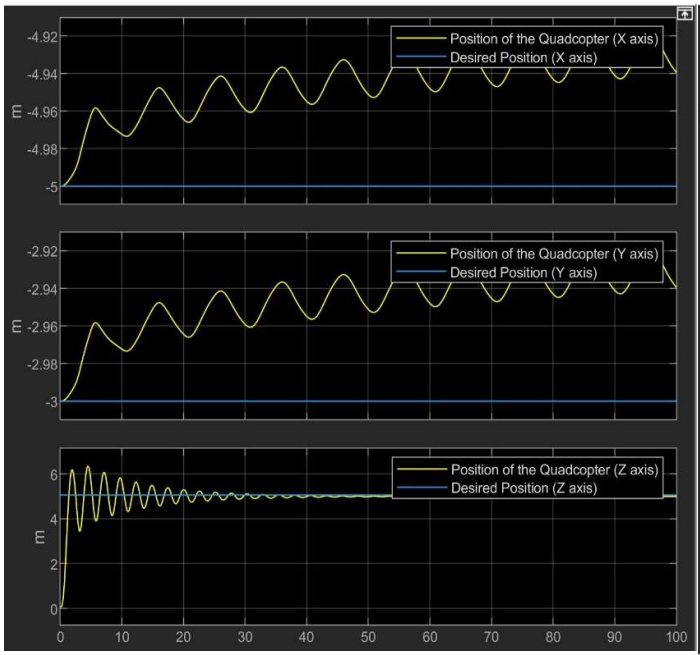


Figure 62 Position Small Wind Simple Feedback

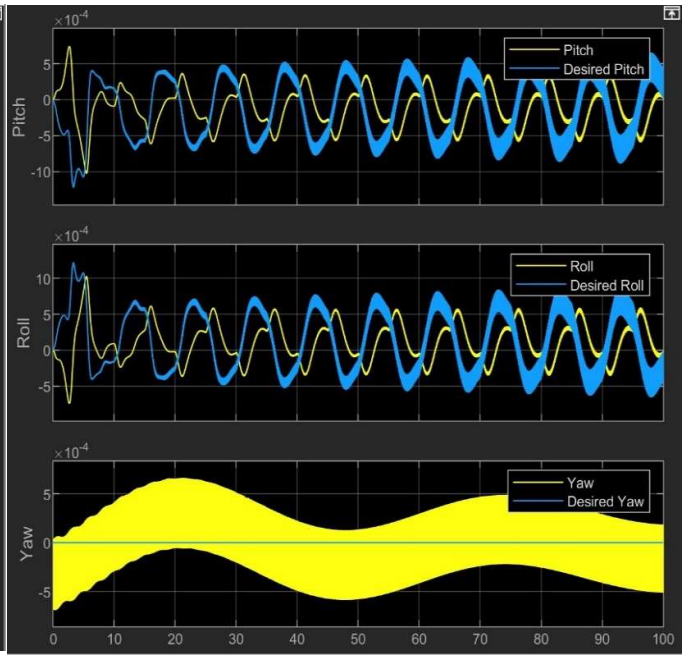


Figure 63 Attitude Small Wind Simple Feedback Control

In this case, simple feedback control and wind disturbance are added to the quadcopter. This simple feedback control uses only a proportional block and works similarly to the ON-OFF controller. When wind disturbance is given the quadcopter is blown away, but it tries to achieve the desired altitude and position. In this, a wave motion can be observed.

4.5.6 PID Control in Small Wind Disturbance

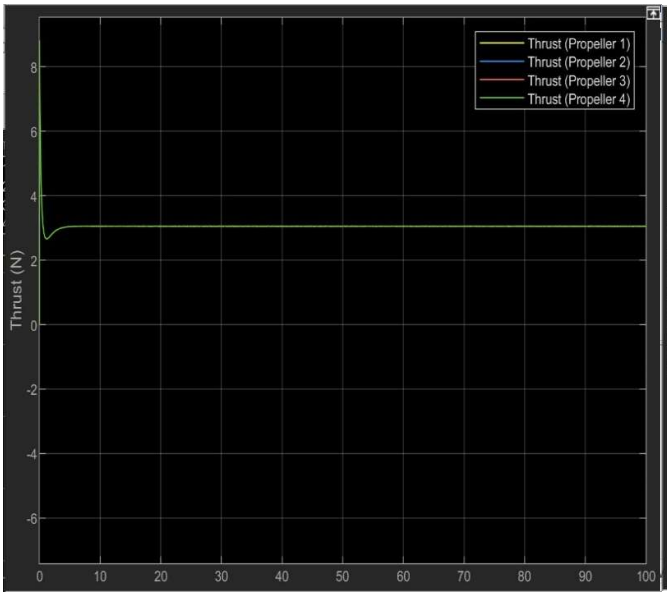


Figure 64 Propeller Thrust Small Wind PID Control

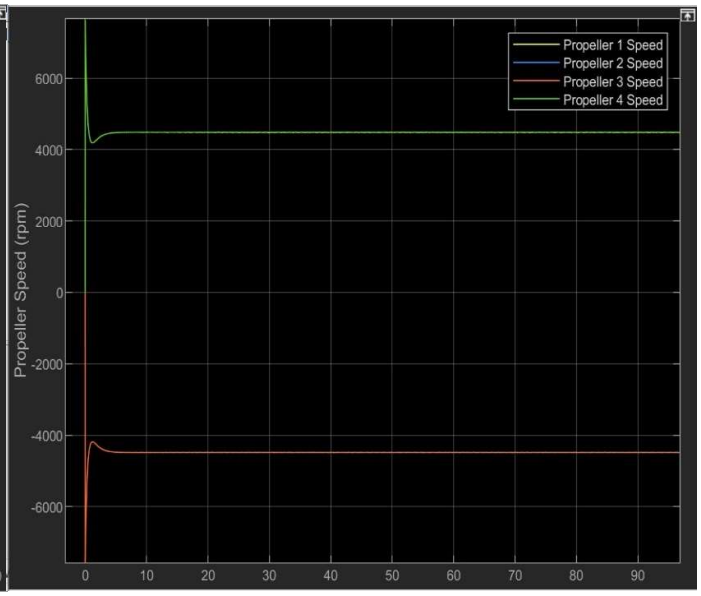


Figure 65 Propeller Speed Small Wind PID Control

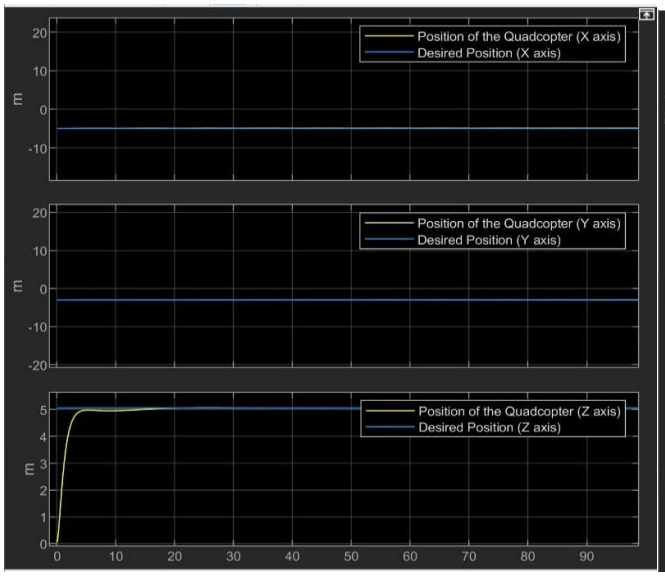


Figure 66 Position Small Wind PID Control

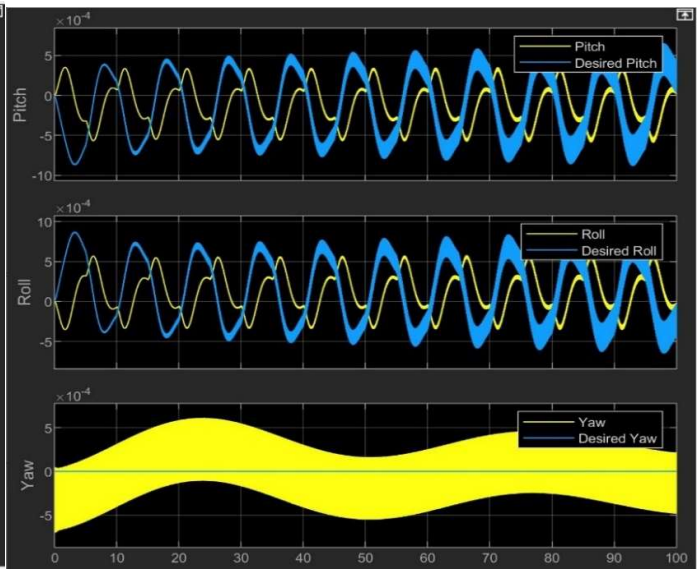


Figure 67 Attitude Small Wind PID Control

A PID controller is implemented in this case. The quadcopter tries to maintain the desired altitude and position and has a smooth motion when compared to other models. For this case, the roll, pitch, yaw axes are also controlled and maintain desired values. Initially when the wind is blown the quadcopter is blown away from its position but using a PID controller it attains the desired X, Y, Z positions by also changing the roll, pitch, yaw angles.

4.5.7 No Control in Large Wind Disturbance

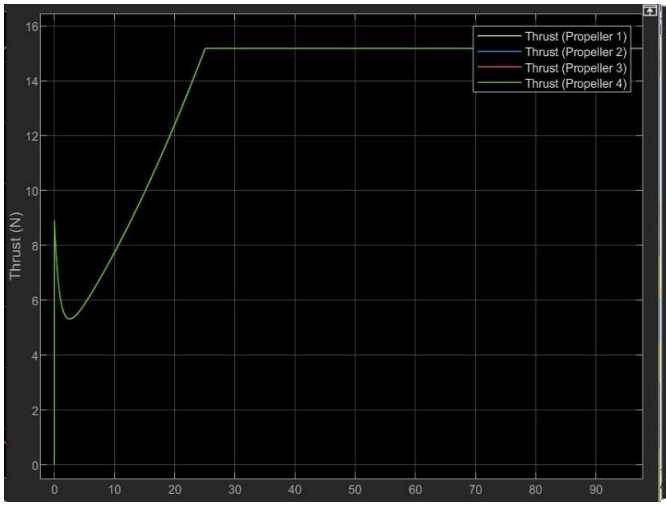


Figure 68 Propeller Thrust Large Wind No Control

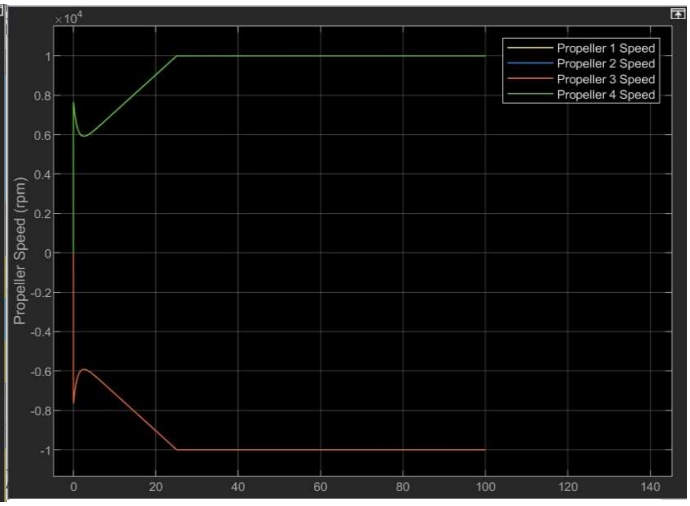


Figure 69 Propeller Speed Large Wind No Control

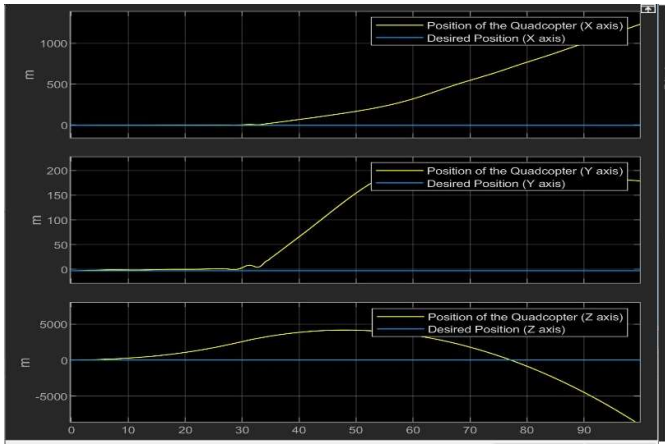


Figure 70 Position Large Wind No Control

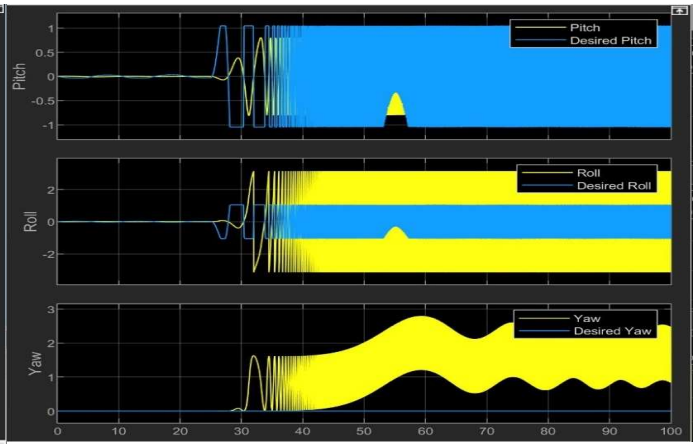


Figure 71 Attitude Large Wind No Control

In this case, as a large wind disturbance of 5m/s is given in the southwest direction

the quadrotor flies away as no altitude control is present and the desired altitude and position are never achieved.

4.5.8 Simple Feedback Control in Large Wind Disturbance

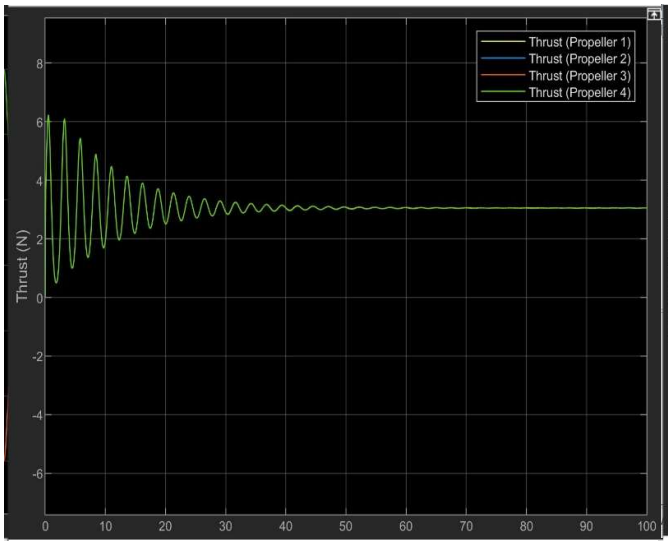


Figure 72 Propeller Thrust Large Wind Simple Feedback

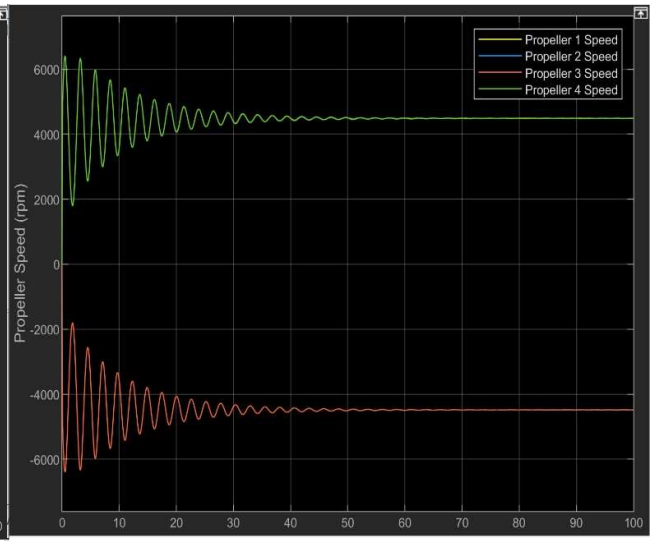


Figure 73 Propeller Speed Large Wind Simple Feedback

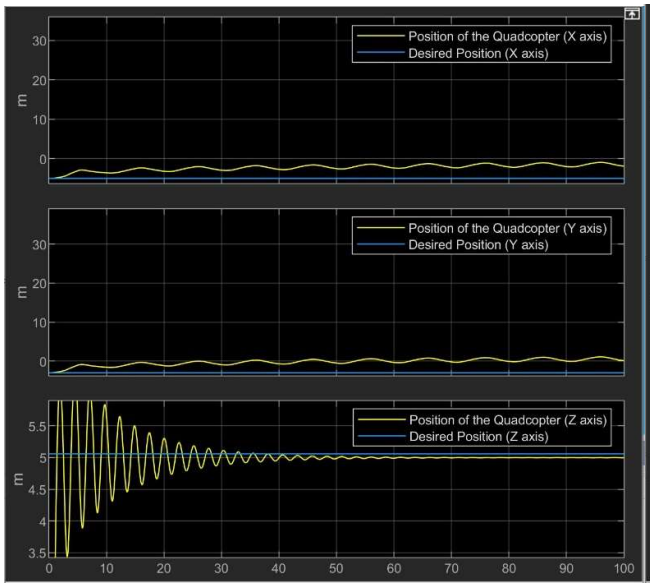


Figure 74 Position Large Wind Simple Feedback

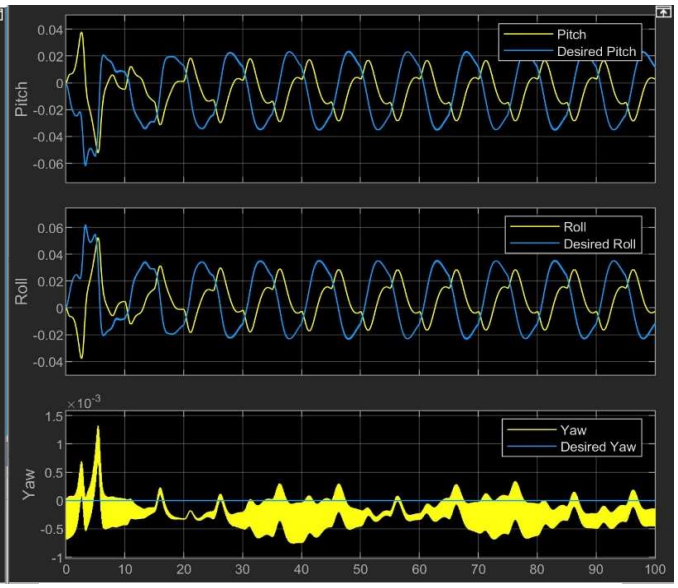


Figure 75 Attitude Large Wind Simple Feedback

In this simple feedback control when a large wind disturbance of 5m/s is given in the southwest direction the quadrotor flies away but the feedback controller tries to

achieve the desired altitude but as this controller works on the ON-OFF principle the motion of the quadcopter is wavy and tries hard to achieve the desired altitude and position but takes more time to achieve due to large wind disturbance

4.5.9 PID Control in Large Wind Disturbance

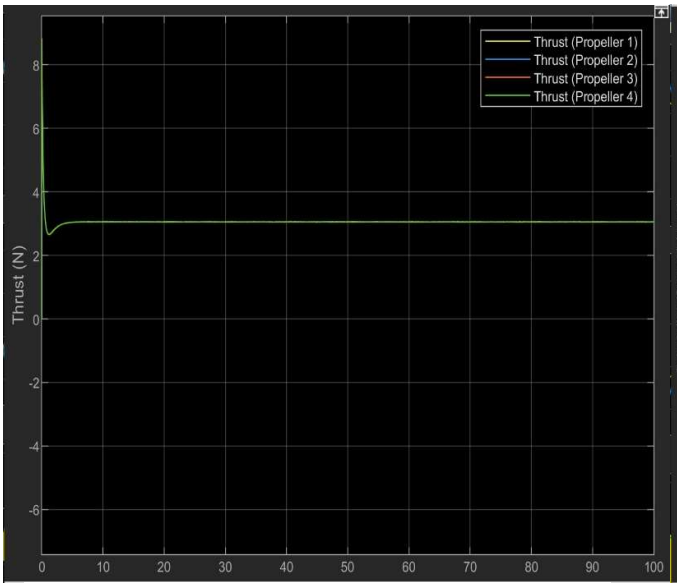


Figure 76 Propeller Thrust Large Wind PID Control

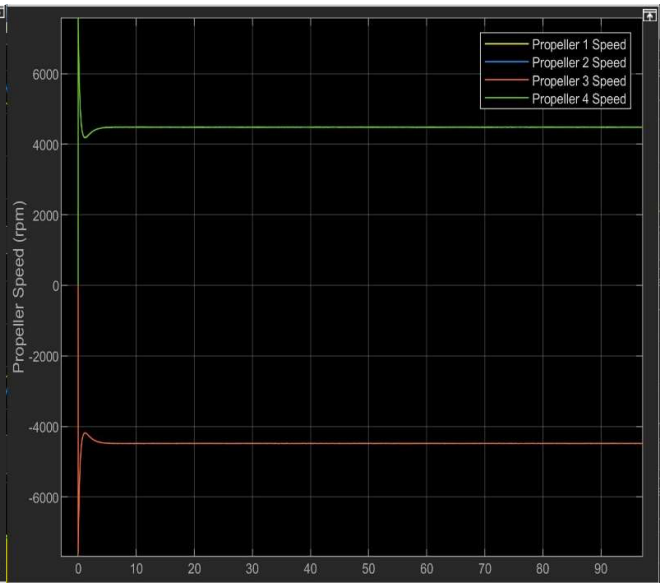


Figure 77 Propeller Speed Large Wind PID Control

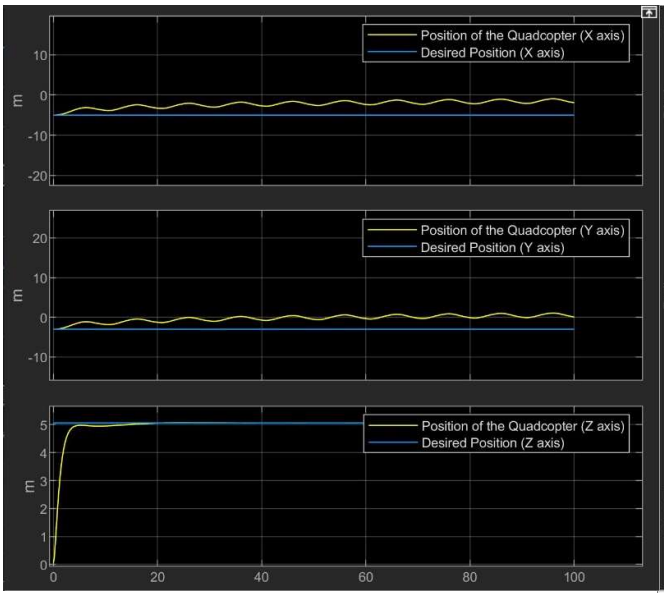


Figure 78 Position Large Wind PID Control

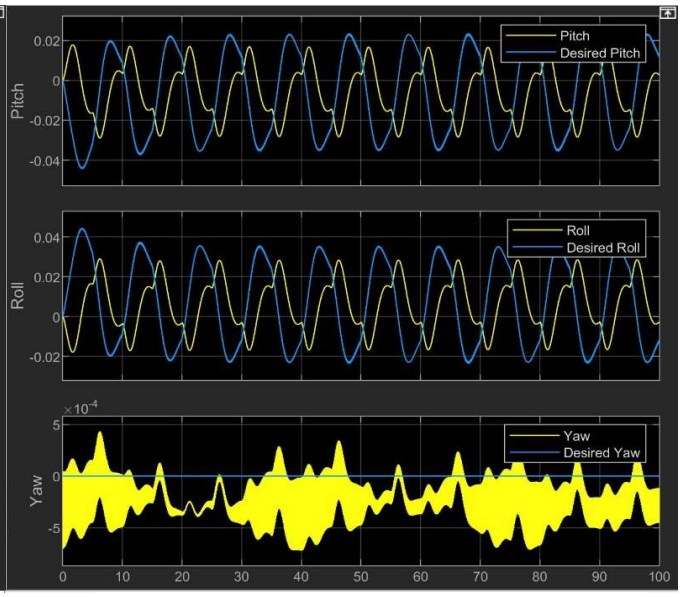


Figure 79 Attitude Large Wind PID Control

By using a PID controller the quadcopter achieves the desired position and altitude even when large wind disturbance is applied, and also the motion of the quadcopter is smooth. Though initially, the quadcopter flies away from the reference frame due to high winds the PID controller adjusts the roll, pitch, yaw angles and achieves the desired altitude, X, Y, Z positions.

4.5.10 PID Control in Snowstorm Conditions

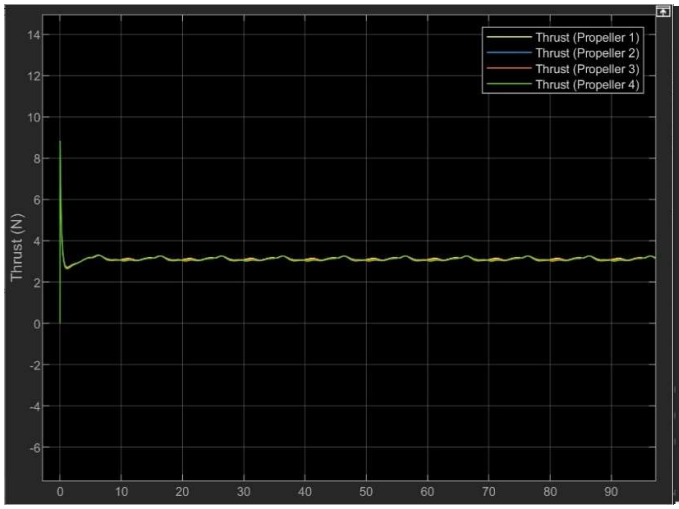


Figure 80 Propeller Thrust Storm Conditions PID Control

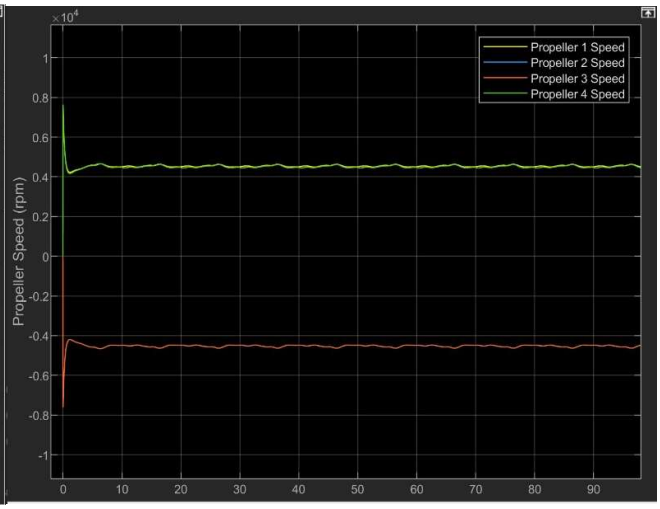


Figure 81 Propeller Speed Storm Conditions PID Control

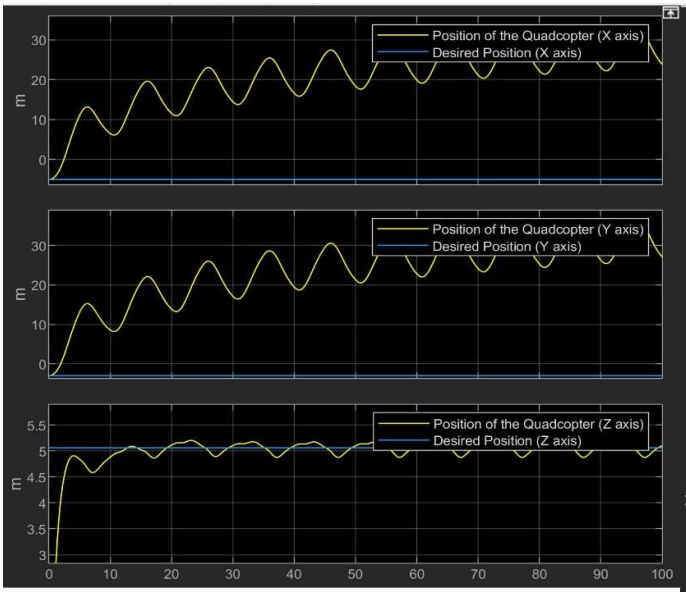


Figure 82 Position Storm Conditions PID Control

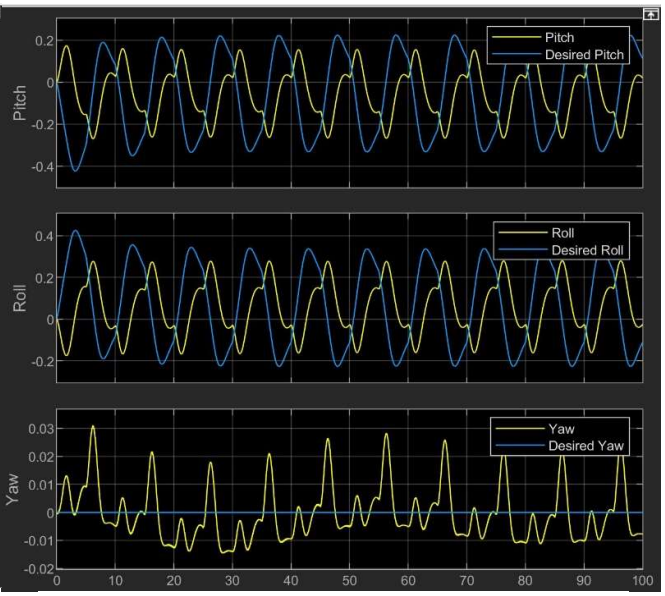


Figure 83 Attitude Storm Conditions PID Control

In this large wind speed of 35 miles per hour which is similar to a typical windstorm, conditions are simulated. Only, the PID controller is implemented. In these conditions, the quadcopter is moving away from the reference frame and can achieve the desired altitude but takes more effort and time as the winds are blowing at a very high speed.

Conclusion

The project was successfully finished, Quadcopter was simulated and controlled properly in MATLAB, Wind disturbance was correctly modeled, Control methodology and quadcopter dynamics were well elaborated. Results, discussion, and analysis were provided.

During this precious and valuable project experience, Strong MATLAB, Simulink, and Simscape programming skills were gained. The use quaternions for rotation instead of direction cosine matrix was learned, Quadcopter dynamics and its PID control theory were learned.

Reference

[1]Z. He and L. Zhao, "A Simple Attitude Control of Quadrotor Helicopter Based on Ziegler-Nichols Rules for Tuning PD Parameters", *The Scientific World Journal*, vol. 2014, pp. 1-13, 2014. Available: 10.1155/2014/280180 [Accessed 17 December 2021]. [2]E. Kuantan, T. Vesselenyi, S. Dzitac, and R. Tarca, "PID and Fuzzy-PID Control Model for Quadcopter Attitude with Disturbance Parameter", *International Journal of Computers Communications & Control*, vol. 12, no. 4, p. 519, 2017. Available: 10.15837/ijccc.2017.4.2962 [Accessed 17 December 2021].

[3]J. Lai, H. Zhou and W. Hu, "A New Adaptive Fuzzy PID Control Method and Its Appliance in FCBTM", *International Journal of Computers Communications & Control*, vol. 11, no. 3, p. 394, 2016. Available: 10.15837/ijccc.2016.3.753 [Accessed 17 December 2021].

[4]D. Gautam and C. Ha, "Control of a Quadrotor Using a Smart Self-Tuning Fuzzy PID Controller", *International Journal of Advanced Robotic Systems*, vol. 10, no. 11, p. 380, 2013. Available: 10.5772/56911 [Accessed 17 December 2021].

[5]J. Bannwarth, Z. Chen, K. Stol, and B. MacDonald, "Disturbance accommodation control for wind rejection of a quadcopter", *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2016. Available: 10.1109/icuas.2016.7502632 [Accessed 17 December 2021].

[6]MathWorks, Package Delivery Quadcopter, 2021. [Online]. Available: https://www.mathworks.com/help/physmod/sm/ug/quadcopter.html
[7]"What Is MATLAB?", 2021. [Online]. Available: https://www.mathworks.com/discovery/what-is-matlab.html. [Accessed: 20- Dec- 2021].
[8]"Simulink tutorial", Ewh.ieee.org, 2021. [Online]. Available: https://ewh.ieee.org/r1/ct/sps/PDF/MATLAB/chapter8.pdf. [Accessed: 20- Dec- 2021].
[9]"What is Simscape", 2021. [Online]. Available: https://www.mathworks.com/products/simscape.html. [Accessed: 20- Dec- 2021].
[10]P. Corke, "robotics-toolbox-matlab/sl_quadrotor.slx at master · petercorke/robotics-toolbox-matlab", GitHub, 2021. [Online]. Available: https://github.com/petercorke/robotics-toolbox-matlab/blob/master/simulink/sl_quadrotor.slx.
[11]"11.7 Performance of Propellers", Web.mit.edu, 2021. [Online]. Available: https://web.mit.edu/16.unified/www/FALL/thermodynamics/notes/node86.html.
[12]F. Ahmad, P. Kumar, R. Dobriyal and P. Patil, "Estimation of the thrust coefficient of a Quadcopter Propeller using Computational Fluid Dynamics", IOP Conference Series: Materials Science and Engineering, vol. 1116, no. 1, p. 012095, 2021. Available: 10.1088/1757-899x/1116/1/012095
[13]E. Fresk and G. Nikolakopoulos, "Full quaternion based attitude control for a quadrotor", 2013 European Control Conference (ECC), 2013. Available: 10.23919/ecc.2013.6669617
[14]A. Gibiansky, "Quadcopter Dynamics and Simulation - Andrew Gibiansky", Andrew.gibiansky.com,2021.[Online].Available: https://andrew.gibiansky.com/blog/physics/quadcopter-dynamics/.
[15]T. Luukkonen, "Modelling and control of quadcopter", Mecharithm.com, 2021. [Online].Available: https://www.mecharithm.com/wp-content/uploads/2020/11/quadrotor_dynamics.pdf.
[16]"Modeling Vehicle Dynamics - Quadcopter Equations of Motion - Autonomy in Motion", Autonomy in Motion, 2021. [Online]. Available: https://charlestytler.com/quadcopter-equations-motion/.
[17] "Features of practical PID controllers", Online-courses.vissim.us, 2021. [Online]. Available: http://www.online-courses.vissim.us/Strathclyde/features_of_practical_pid_contro.htm.

----------X----------