Addis Ababa University
Addis Ababa Institute of Technology
School of Electrical and Computer Engineering

Particle Swarm Optimization Tuned Fractional Order Sliding Mode Controller for Altitude Stabilization and Trajectory Tracking of Agricultural Monitoring Quadcopter.

A thesis submitted to School of Graduate Studies, Addis Ababa Institute of Technology, Addis Ababa University in partial fulfilment of the requirement for the Degree of Master of Science in Electrical Engineering (Control Engineering)

By

Belsty Derseh

Advisors

Dr.Lebsework Negash and Dr.Dereje Shiferaw

November 3, 2021

Addis Ababa, Ethiopia

Addis Ababa University

Addis Ababa Institute of Technology

School of Electrical and Computer Engineering

Particle swarm optimization Tuned Fractional Order Sliding Mode Controller for Altitude Stabilization and Trajectory Tracking of Agricultural Monitoring Quadcopter.

By: Belsty Derseh

APPROVED BY BOARD OF EXAMINERS

Name	Signature	Date
Dr.Bisrat Derebssa (School dean)	.....	.....
Dr.Lebsework Negash and Dr.Dereje Shiferaw (Advisors)	.....	.....
Dr.Chala Merga (Internal Examiner)	.....	.....
Mr.Yared Tadesse (External Examiner)	.....	.....

Declaration

I declare that the work entitled "PSO tuned FOSMC for altitude stabilization and trajectory tracking of Agricultural monitoring UAV" is my original work and has not been presented for the fulfillment of any degree in this university or any other university or colleges, as well as all sources of material used for the thesis have been duly acknowledged.

Name	Signature	Date
Belsty Derseh	.....	.....

Dedication

To my father Derseh Meheretiea, my mother Enatihun Gashu, my brothers ,sisters and my friends.

"Family: is the beginning and the end."

Belsty Derseh

Acknowledgment

First and Foremost, I sincerely look-up-to The Almighty God for His Grace, for giving me strength, sustenance and above all, for his fatherhood and Love from the beginning of my academic life upto this Masters Level. God's benevolence has made me Excel and Successful in all my academic pursuits.

Besides God, I would like to express my deepest gratitude to my advisor's, Dr. Lebsework Negash and Dr.Dereje Shiferaw, for the continuous support on my Msc study and research, for their patience, enthusiasm, immense knowledge and more than anything for believing in my potential. Their guidance helped me in all the time of the research and writing of the thesis.

Finally I would like to extend my special thanks to my parents for their encouragement and emotional support to the success full completion of my thesis and my classmates for their grate suggestions in my thesis work.

Belsty Derseh

Abstract

Quadcopter technology could help farmers around the world to monitor their agriculture to know accurate and up-to-date information quickly on the health of their crops and the environmental condition of the land. This thesis addresses particle swarm optimization tuned fractional-order sliding mode controller for altitude stabilization and trajectory tracking of agricultural monitoring UAV. First, the quadrotor dynamics are modeled using the Newton-Euler systems approach. Second, Fractional-order (FO) sliding mode control (SMC) system is developed for attitude and position trajectory tracking of a quadrotor unmanned aerial vehicle (UAV) system under unknown external disturbance. Quadcopter control algorism is divided into inner and outer control loops because the quadcopter is under actuated system, Where direct control of all six degrees of freedom is not possible. The outer loop controls the altitude and generates roll and pitch angle reference trajectories controlled in the inner loop. The inner loop controls attitude (roll, pitch, and yaw) of the quadcopter.
In order to achieve good task performance for agility, flying efficiency and trajectory tracking, particle swarm optimization (PSO) algorithms are used to obtain parameters of fractional order sliding mode controller (FOSMC).
The comparison of conventional sliding mode control with fractional-order sliding mode controller is analyzed. The effectiveness of the proposed control scheme is verified by developing simulation results for the quadcopter study in MATLAB/SIMULINK software. In this work, different types of tasks are performed under different conditions. Indeed, the ability of the proposed controller to track the imposed trajectories and achieving of position in space is well seen from 3D path tracking simulations and even in presence of disturbances.

KEY WORDS: - quadcopter, particle swarm optimization, fractional order sliding mode controller, Trajectory, Disturbance

Contents

List of Figures

List of Tables

List of Abbreviations

CCW	Counter clockwise
CTOL	Conventional Take-off And Landing
CW	Clockwise
FOC	Fractional Order Calculus
FOSMC	Fractional order Sliding Mode Controller
ISE	Integral Square Error
ITAE	Integral Time Absolute Error
ITSE	Integral Time Square Error
PID	Proportional Integral Derivative
PSO	Particle swarm Optimization
SMC	Sliding Mode Controller
UAV	unmanned Aerial Vehicle
VTOL	Vertical Take-off And Landing
WRT	With Respect To

Chapter 1

Introduction

1.1 Background

UAV, an abbreviation of Unmanned Aerial Vehicle, sometimes also referred as Unpiloted Aerial Vehicle, which is a self-powered flying object without intervention of human pilot. UAV have been widely used to facilitate human life. Quadcopters are used in various fields ranging from the military, humanitarian relief to agriculture. In recent decades, agriculture is the most important industry in world. Agricultural science and technology pays high attention to output of various agricultural products and enhancing the transformation from traditional to modern agriculture. Agricultural monitoring to become one of the primary methods of quickly and accurately obtaining agricultural information. These vehicles are used in agriculture for a number of missions including separating weeds from crop, capturing image for analysis of individual plant leaves, mapping of an unknown environment, automatic monitoring of forest fires and management of forest health etc. These vehicle have a huge potential in agriculture in supporting evidence-based planning and in spatial data collection.

Quadcopters are generally classified into two categories. Remotely piloted aircrafts and Autonomous aircrafts. In both categories, the common thing is that there is no human directly involved or present in the flight environment. In the Remotely piloted aircraft, the UAV is control and guide through the way by humans over some sort of wireless communication. However, in the Autonomous aircrafts category the vehicle can intelligent enough to fly and perform the designated task by itself. Two species of UAVs are.

- Rotary wing variant.
- Fixed-wing variant

Rigid wing capable of generating lift. In order to fly, fixed-wing UAVs must have an engine to generate thrust force used to moves the craft along the ground while the greater air

pressure generated below the wing generates lift. The other basic division in UAV Classification is based on their landing and take-off abilities are Conventional Take-off and Landing (CTOL) capability and Vertical Take-off and Landing (VTOL) capability. Now a day, most of the commercially used UAVs are of the VTOL category. VTOL provides the aircraft to land and take-off in places where conventional vehicles cannot, But CTOL category UAVs are popular for their speed and high load carrying capabilities. However, research and specific application demands make VTOL aircrafts the ultimate choice. Among this class of UAVs, Multi-rotor UAVs are the most current ones because of their higher stability in flight and easy control. Quad-copter is also a type of rotary wing Multi-rotor VTOL type UAV. The advancement of quad-copters has challenged until recent time, because controlling four independent rotors has difficult and impossible without electronic assistance.
Quadcopters have four rotors arranged at the edge of a cross body allowing them to move at different speeds. Two diagonal rotors rotates a clockwise rotation while the other two diagonal rotors rotates anti-clockwise direction. This configuration is important because it keeps the quadcopter balanced by generating equal in magnitude and opposite directional torque . If all the motors were to rotate in the same direction, the resulting torque is maximum and in one direction, which make the whole craft rotate. The rotors are designed in such a way that they push down air generating lift. When the rotors push down air, the air pushes back on them propelling the drone upwards. The faster the rotors spin, leads to the faster the drone upward motions. When rotors spin slow down, the drone starts to move downward to ground. The movement of a quadcopter relies on the rotation of the four rotors to generate thrusts and to control its attitude and position.

Figure (1.1) Quad-copter Model [1]

1.2 Problem Statement

The quadcopter has four rotors that are controlled independently. The movement of the quadcopter results from changes in the speed of the rotors. The quadcopter is naturally

unstable, non- linear and being under-actuated. In order to achieve the desired trajectory, design six degrees of freedom by coupling three translational and three rotational motion. The automatic control of a quadcopter is not a straight forward due to its under-actuated properties, it is difficult to control all these six states with only four control inputs. Because of quadrotor is sensitive to external disturbances and parameter variations easily, the FOSMC is used as a nonlinear controller to combine the flexibility of the fractional-order approach to keep quadrotor on desired trajectory, as well as disturbance rejections and error minimization. The fractional-order calculus (FOC) is very attractive method to obtain more sensitive control achievement compared to integer.
Designing a controller for quadcopter is somewhat complex because tuning of the FOSM controller parameters gain is another challenging problem for flying efficiency and trajectory tracking . To deal with such a difficulty, PSO algorithms are used to obtain parameter gains of Sliding Mode Controller (FOSMC). In this paper, the controller that is proposed to control altitude stabilization of quadrotor in order to track the desired trajectory and disturbance rejection by fractional order sliding mode controller (FOSMC) tuned with particle swarm optimization (PSO). Therefore, the proposed controllers with proper design will be able to overcome the above-mentioned challenge

1.3 Objective

1.3.1 General Objective

- To improve the performance of altitude stabilization and trajectory tracking of agricultural UAV by using PSO tuned fractional order sliding mode controller.

1.3.2 Specific Objectives

- To model and simulate an agricultural monitoring UAV that can be used to monitor agricultural fields.
- To design a FOSM controller tuned with PSO for an agricultural monitoring UAV for efficient stabilization and trajectory tracking.
- To evaluate the performance of the designed controller in terms of trajectory tracking and stabilizing in the presence of disturbances.
- To simulate the mathematical models with the proposed controller using simulation in MATLAB/SIMULINK.

1.4 Significant of the study

In the past few years, Quadcopter has obtained a significant interest in both military and civilian applications. These vehicles are used in agriculture for several missions including separating weeds from the crop, capturing an image for analysis of individual plant leaves, mapping of an unknown environment, automatic monitoring of forest fires, management of forest health, obtaining information on soil water holding capacity or management irrigation systems especially for large agricultural producers that cultivate in regions with dispersed areas, etc. Apart from favorable climatic conditions for growing vegetables, our country still has no leading technical facilities necessary for the competitiveness of the market economy. Thus, any solution by which farmers can increase the product quantities; reduce production costs while maintaining product quality and integrity within minimum energy is considered. As the usage of these vehicles becomes widespread, the development of controller structures, which allow the UAVs to follow a specified trajectory within minimum error and unaffected by external disturbance precisely, is a new area of interest for researchers.

1.5 Methodology

The study will be achieved through the following methodology:

- Literature Survey: Start with reviewing of literature, which includes reading of books, publication papers and thesis works to get essential information, concept and ideas that will assist to focus on the thesis.
- Obtaining the mathematical model of the quadcopter.
- Fractional order sliding mode control has been designed which is robust and insensitive to external disturbances, un modelled dynamics and parameter uncertainties.
- Develop Particle swarm optimization technique that has automatic tuner proven an effective and efficient method of gain tuning to obtaining or select the parameters of the controller gains.
- MATLAB/Simulink Simulation has been utilized for comparison of trajectory tracking performance for convectional sliding mode control with fractional order sliding mode control tuned with PSO and validating the mode
- PSO tuned fractional order sliding mode control has been tested subjected to different effects to ensure the robustness, parameter uncertainty and insensitive to external disturbance while tracking the desired reference trajectory of quadcopter.

1.6 Scope of the thesis

The scope of this thesis is develop mathematical model, design and simulation of fractional order sliding mode controller tuned with PSO for efficient stabilization and trajectory tracking in the presence of external disturbances for the dynamic model of quadcopter. The design of the system and controller will be implemented on MATLAB/Simulink environment.

1.7 Outlines of the thesis

The thesis is organized into six chapters including the introduction in chapter 1. The rest of the thesis is organized as follows.

1. **Chapter 2:** Describes the literature review, the basic structure (alignment) and flight principles of Quadcopter.
2. **Chapter 3:** Deals with mathematical modelling of the quadcopter design.
3. **Chapter 4:** Design fractional order sliding mode controllers' and tune with PSO.
4. **Chapter 5:** Presents the results obtained along with a discussion on those results briefly.
5. **Chapter 6:** Draws the conclusion from the work done in this thesis adding recommendation and future works to be done.

Chapter 2

Overview of quadcopter

2.1 Literature review

The quadcopter has become one of the most popular unmanned aerial vehicles (UAVs), which attract considerable attention in both academia , industry and agricultural sectors. Compared with traditional manned air-plane and unmanned fixed-wing flight vehicles, quadrotor has its unique advantages, such as low cost, small size, and hovering and vertical take-off landing, simple structure and flexible flight ability. As a result, quadcopter is playing a significant role in agricultural,the advancement of quadcopter and achieving high-performance controller has challenged issue until recent time, because controlling four independent rotors has difficult,motivate and interesting problem .

Different literatures have approached the research of quad-rotor UAV in different ways. The quadcopter's control problem has been widely investigated using several control approaches and some of them are discussed here below.

Derafa [3] , formulated the equations of motion that govern the dynamics of the craft mathematically using the Newton-Euler mechanics. PID controller has been used to control the yaw angle and the altitude on a square wave trajectory. It is recommended that the other DoF determining its position in space be also stabilized for a better outcome in controlling the motion of the vehicle. The four-rotor control system was built for only attitude angles and height along z-axis and verify the model outputs.

Axel Reizenstein[4], design and implement controllers of the quadcopter's position and trajectory in an outdoor environment. Having two controller's i.e. inner and outer .outer loop controllers will generate reference angles for the inner loop angle controllers. PID and LQG controller has been used to control by linearizing the dynamics and tuning manually. Implementing an optimal and non-linear control algorithm to decrease trajectory error and steeling time to follow the desired position is his draw back for further work.

In the paper by Ruth Tesfaye[5], Modelling and Control of a Quad-rotor Unmanned Aerial Vehicle at Hovering state only. Linearize the model and introduce disturbances to stabilize craft back to hovering position . Finally compare PID and LQR controller to stabilize the quad-rotor in the hovering position.
Tommaso Bresciani,[2],also derives the mathematical model of the quad-rotor using the Newton Euler Formulation. Accurate description of the quadrotor components and their interconnections are experimentally investigated to develop a real platform.To improve this quadrotor parameters in simulation environment PID controller has been used to stabilize height and attitude with two levels. i.e, the low-level controller and the high-level controller. The low-level controller's for the purpose of stabilization of the height and attitude whereas the high-level controller is cascaded with the previous controller to follow for position requirements, obstacle avoidance and trajectory tracking.
The authors in [6] presented a similar work to those in [2] the mathematical model of kinematics and dynamics of the vehicle was Implemented,PID controller built upon and tested on an Arduino hardware for data collection and control system evaluation. Internet was the primary source of information and controller was not designed to maintain the take-off position of the vehicle. Further PID tuning is required for improved flight performance.
Data analysis of quadcopter dynamic attitude on a circular trajectory and comparing the modeling results of conventional Proportional Integral Derivative (PID) and Fuzzy-PID controllers investigated in[7]. Manual controller gain adjustment has a crucial role in quadcopter flight.
Many of those studies use proportional-integral-derivative (PID) controller, which is easily applied in UAV for its practicability but not robust. The challenging problem is finding the optimal parameters of PID controller.The above researcher presents using manual adjustment and Ziegler-Nichols rules for tuning controller parameters of quadrotor . The authors in [8] proposed tuning method for determining the parameters of PID controller using PSO.

Particle swarm optimization (PSO) algorithm is a new concept that has proven to be an effective and efficient method of gain tuning for a number of systems and controller types [8] [9] [10]. The authors in [11] states Sliding Mode Control for Quadcopter altitude and attitude stabilization is presented and tuned using Particle Swarm Optimization. PSO and CS is used to improve results concerning the quadrotor full control. However, the CS algorithm suffers from a low convergence speed.
Particle swarm optimization (PSO) algorithm tuned optimal PID gains proposed approach is not highly robust and not very sensitive in term of changing of the weighted constants [12]. [13] author specifies Particle swarm Optimization in application for quadrotor attitude

and path following control tuning using feed forward plus pd controller. To achieve a desired performance for agility, flying efficiency and immediate reaction is a challenging problem. Adjusting the gain parameters and implementing the PID controller is still easy and it yields not only reliable results but also gives satisfactory results for linearized dynamic models. All researchers mentioned above use the linearized model of the quadcopter. However, quadrotors have highly nonlinear dynamics, parametric uncertainties and affected by external disturbance such as wind gusts, rains, snows and sensor spoofing. Due to these distorting effects and properties, it is necessary to acquire perfect results especially if the tracking performance needs to be precise. The researchers have been working until recent time to control uncertain dynamic systems under external disturbances. The Sliding Mode Control (SMC) is one of the robust, nonlinear control method, which drives the phase trajectory to a predetermined sliding surface and switches on it. SMC has found wide usage in literature.
Research presented in [14][15] ,applies SMC to adopt the problem of attitude control with external disturbances using the nonlinear sliding mode control. This approach has the ability to stabilize the quadrotor and move it to any desired position with an appropriate switching function even despite an injected noise, but there is a serious of chattering problem and gains of the controller are adjusted manually. To eliminate chattering problem that is frequently encounter in standard SMC numerous methods were proposed,[16] Adaptive Super-twisting Second-order Sliding mode control (ASTSOSMC) is designed and implemented in real time for the altitude tracking of a quadrotor aircraft.

In the literature by [17][18] , a fractional-order sliding mode controller (FOSMC) is designed and applied to a quadrotor for trajectory tracking control at hovering state. The experimental results show that the FOSMC is more robust to parameter uncertainty and better than SMC in terms of reference tracking, error percentages and gives fast response to the changes in the references.but they use tradetional gain tunning methods.
In [19], Adaptive Sliding Mode Controller (ASMC) describes that the control laws can eliminate the altitude and attitude tracking errors and guaranteed to converge to zero asymptotically, even under a strong external disturbance. The Comparison with LQR, ADRC and the proposed ASMC has been carried out. Robustness of the controller under un modelled dynamics and external disturbances is assured [8].However, in all case the controller gains are obtained by trial and error method.
In this paper, PSO tuned fractional-order sliding mode controller (FOSMC) is designed and applied In order to enhance trajectory-tracking performance of the quadrotor the attitude stabilization problem of under-actuated quadrotor aircraft in presence of external disturbances is considered. Particle swarm optimization has automatic tuner proven an effective and efficient method of gain tuning.

2.2 Flight planning of agriculture monitoring UAV

Agriculture is the most important industry in the world. In recent decades, agricultural science and technology pays high attention to the output of various agricultural products and enhancing the transformation from traditional to modern agriculture. Agricultural monitoring to become one of the primary methods of quickly and accurately obtaining agricultural information. The objective of this thesis is to design a controller used to control the position and attitude stabilization of the quadrotor to track the desired trajectory to monitor agricultural maps of the area given in figure e 2.1.

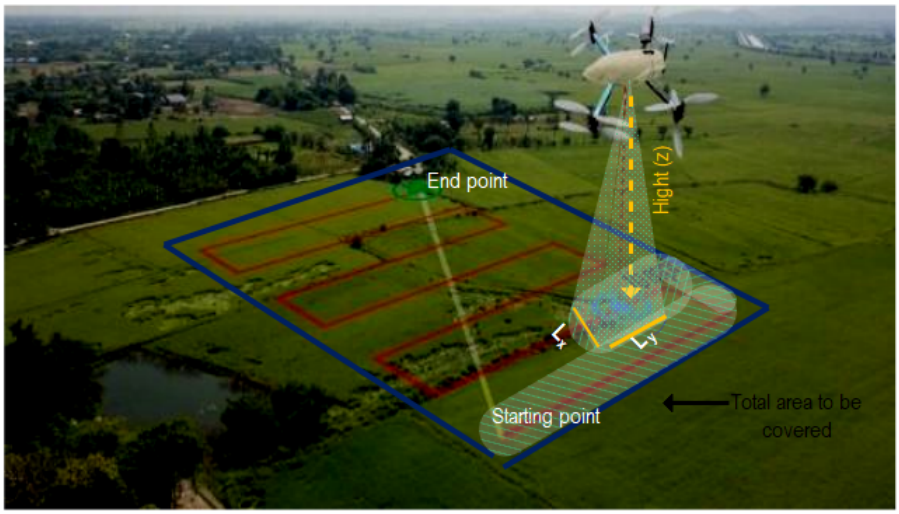


Figure (2.1) Square wave trajectory of UAV for agriculture monitoring.

The size of the image on the ground is directly proportional to the altitude of the UAV above the ground.

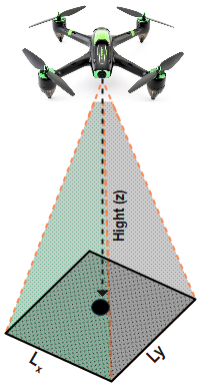


Figure (2.2) Area projected by a camera with the angle of view α flying at altitude Z.

The size (L_x, L_y) of the ground image taken by the UAV flying at height Z above the ground depends on altitude Z and angle of view of the camera.

$$\begin{aligned} L_x &= 2Z\tan(\frac{\alpha}{2}) \\ L_y &= \frac{L_x}{\rho} \end{aligned} \tag{2.1}$$

Where L_x is the width of the image on the ground, L_y is the length of the image on the ground, α is the angle of view of the camera, ρ is the aspect ratio ($\rho = I_x/I_y$), I_x is the width of the image expressed in pixels, and I_y is the length of the image expressed in pixel.

2.3 The basic structure of the four-rotor aircraft

The layout of a quad-copter with two cantilevers, each with a rotor. Motors 1 and 3 are mount on a similar arm, turn a counter-clockwise way (CCW) while motors 2 and 4 are mount on the subsequent arm, and pivot in the clockwise direction (CW). The two motors should rotate in the same direction aligned at the opposite ends of the same arm as shown in figure 2.3. This configuration is used to prevent torque imbalance during linear flight.

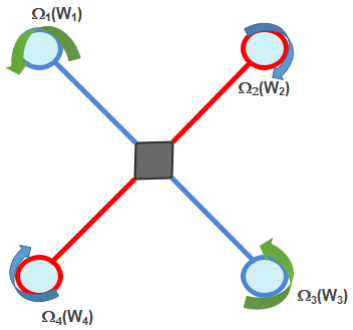
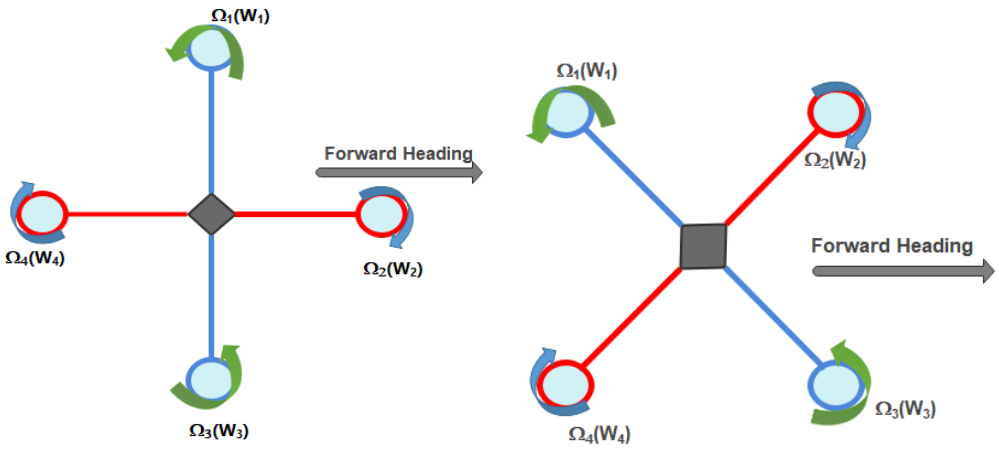


Figure (2.3) Structure of UAV

2.3.1 Types of quad-copter Configuration

A quad-copter has two main configurations, which are common [20]. The first is $''plus(+)''$ configuration, where a single rotor leads the air. Where as the $''cross(x)''$ configuration, two rotors lead the air craft.



(a) Plus configuration (b) cross configuration

Figure (2.4) Plus and cross configuration of quadcopter

For the following analysis, the cross configuration will be used because of the following two main advantages:-

- The cross configuration provides more stability than plus configuration,i.e. if one of the rotors failures occurs, the quad-copter moves stably due to the second rotor in case of cross configuration.
- The cross configuration guarantees no camera arc obstruction for a camera installed on a quadcopter.

Assumptions Taken

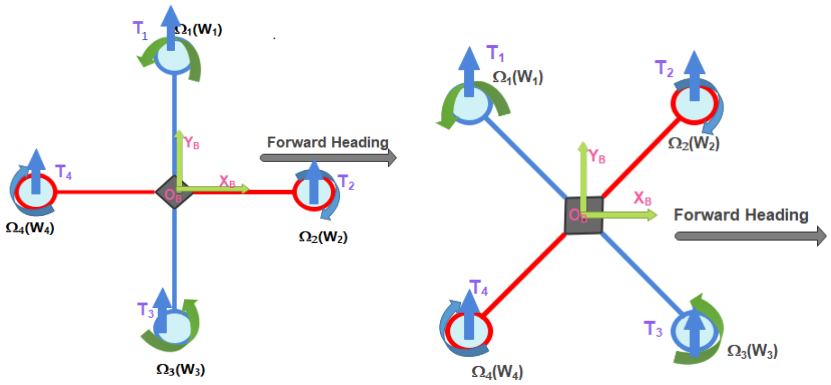
- The quad-copter is a rigid structure and has a symmetric.
- Thrust is proportional to the square of rotor speed ($f_i = b\Omega_i^2$).
- Torque is proportional to the square of rotor speed ($\tau_i = d\Omega_i^2$).
- The center of mass of the quadcopter and the origin of the body frame coincide.

2.3.2 Quadcopter flight principles with basic forces

There are four basic movements, which allow the quad-copter to reach a certain height and attitude (orientation) [21].

1. **Thrust:-**It is performed by increasing the speed of all four rotors simultaneously with the same magnitude. It leads to a vertical force with respect to the body reference

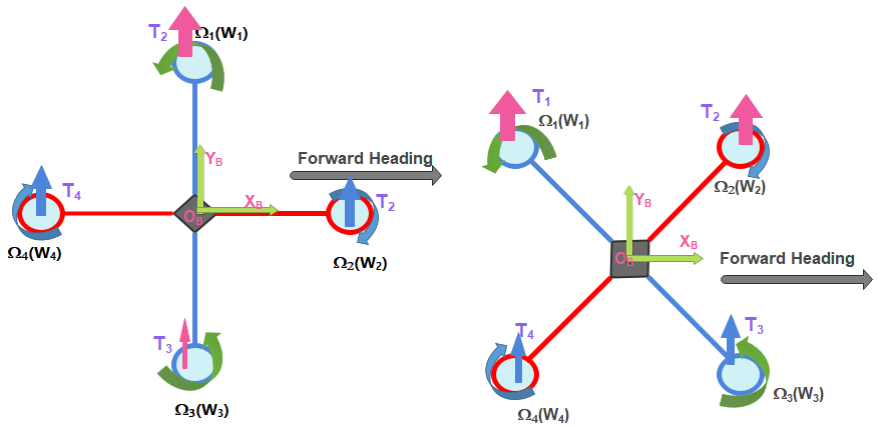
frame that increases or decreases the altitude of the quadrotor. Motors 1 and 3 rotate counter-clockwise, and motors 2 and 4 rotate clockwise. Since the rotation speed of the four motors is equal, the rotor produces the same lift force ($T_1 = T_2 = T_3 = T_4$). The torque produced in each arm is equal in magnitude but opposite in direction, so the net torque just cancels out each other and then stabilizes vertically. At this time, if the rotation speed of four motors increased simultaneously, the lift force of the body will increase. When the sum of the force of the four rotors is greater than the gravity of the body, the quad-copter will take off vertical upward motion. When the sum of the force of the four rotors is less than the gravitational force of the body, the quad-copter will go in a vertical downward motion.



(a) Trust for Plus configuration (b) Trust for cross configuration

Figure (2.5) Plus and cross configuration for trust movement of quadcopter.

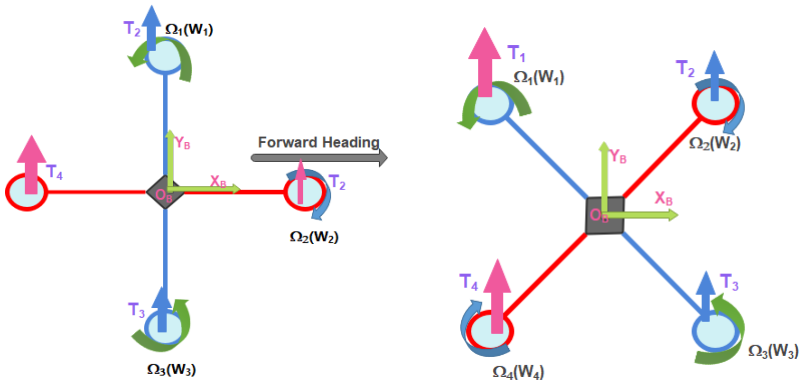
2. **Roll:-** The plus configuration is performed by increasing or decreasing the speed of the rotor on the left $\Omega_1(W_1)$ and by decreasing or increasing the speed of the rotor on the right $\Omega_3(W_3)$, which uses only two rotors to roll. While the cross configuration engages all of its rotors to perform rolling as follows. Increasing or decreasing the speed of the rotor $\Omega_1(W_1)$ and $\Omega_2(W_2)$, and by decreasing or increasing the speed of the rotor $\Omega_3(W_3)$ and $\Omega_4(W_4)$, resulting in four-wing aircraft left and right ends of the unbalanced lift so that the quadcopter body flip to the right or left.



(a) Roll for Plus configuration (b) Roll for cross configuration

Figure (2.6) Roll movement Plus and cross configuration for of quadcopter.

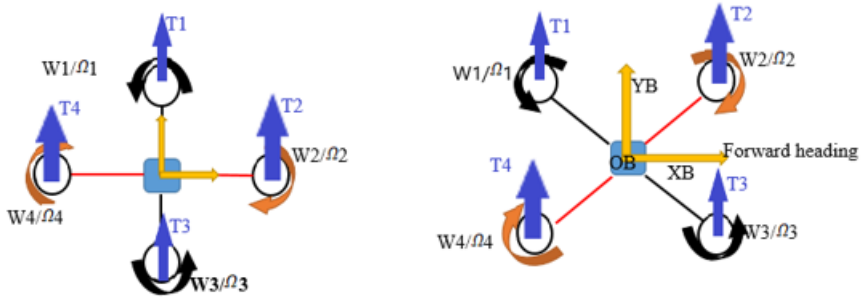
3. **Pitch:-**The plus configuration is performed by increasing or decreasing the speed of the rotor on the rear $\Omega_4(W_4)$ and by decreasing or increasing the speed of the rotor on the front $\Omega_2(W2)$, Which uses only two of its rotors in contrary to the cross configuration which uses all of its four rotors as follows. Increasing or decreasing the speed of the rotor $\Omega_1(W_1)$ and $\Omega_4(W_4)$, and by decreasing or increasing the speed of the rotor $\Omega_2(W_2)$ and $\Omega_3(W_3)$, resulting in four-wing aircraft front and back ends of unbalanced lift, so that the quadcopter body flip to the front and back.



(a) pitch for Plus configuration (b) pitch for cross configuration

Figure (2.7) Plus and cross configuration for pitch movement of quadcopter

4. **Yaw:-**Increasing or decreasing the speed of the rotor on the rear front and decreasing or increasing the speed of the rotor on the left-right. This movement is clockwise or anti-clockwise around the aircraft's central axis of rotation.



(a) yaw for Plus configuration (b) yaw for cross configuration

Figure (2.8) Plus and cross configuration for yaw movement of quadcopter

Remark

- **For the plus configuration** , considering the pitch movement, where a single rear rotor (w4) slows down and a single front rotor $\Omega_2(w_2)$ speeds up it generates nose-up pitching. Since torque does not vary linearly with the speed of a rotor ($\tau_i = d\Omega_i^2$) The decrease in torque of the CW spinning rear rotor does not identically cancel with the torque of the CW spinning front rotor, which leads to net yaw moment. Therefore, the plus configuration requires compensation with a yaw control input. Similarly, considering the roll movement, where a single left rotor slows down and a single right rotor speeds up to generate a roll-left moment. Since torque does not vary linearly with the speed of a rotor ($\tau_i = d\Omega_i^2$) The increase in torque of the CCW spinning right rotor does not identically cancel with the torque reduction of the CCW spinning left rotor, resulting in net yaw moment. Therefore, the plus configuration requires compensation with a yaw control input.

- **For the cross configuration**,considering the pitch movement, where the two adjacent rear rotors $\Omega_1(w_1)$ and $\Omega_4(w_4)$ speed up and the two adjacent fronts $\Omega_2(w_2)$ and $\Omega_3(w_3)$ rotors slow down it generates nose-down pitching. For those two rear rotors, slowing down one rotates in CW direction and the other in CCW direction, and the torque generated cancels out. The same is true for the front rotors speeding up. Thus, the net torque is canceled out and leads to pitch control does not introduce the net yaw moment. Similarly, considering the roll movement, where the two right rotors speed up and the two left rotors to slow down to generate a roll-right movement. From the two right rotors, slowing down or increases one rotor in CW direction and the other in CCW direction, and the torque generated cancels out. The same is true for the left rotors slowing down Thus, the net torque is canceled out and leads to the roll

control does not introduce a net yaw moment in cross configuration compared to plus configuration.

- The rotor thrust not varying linearly with rotor speed ($f_i = b\Omega_i^2$), so the increase in thrust of speeding up rotors does not cancel identically with the reduction in thrust from rotors slowing down by the same amount. Therefore, for both configurations, the pitch, roll, and yaw control modes result in small net changes in thrust, and require compensatory collective control input.

Chapter 3

Methodology

3.1 Modeling Approach

In this chapter, Newton-Euler modelling approaches are used to study the behaviour of the quadcopters. Quadcopters are under-actuated systems in which four independent inputs are used to control motion in six degrees of freedom, three translational and three rotational. Because of aerodynamic effects, the resulting dynamics are highly nonlinear. Quadcopters are either lifted up ward, moves along x-axis, along y-axis, in the negative direction propelled using four vertically oriented propellers. Quadrotors use four identical rotors arranged at the edge of a cross body, Two diagonal rotors rotate a clockwise rotation while the other two diagonal rotors rotate anti-clockwise direction. To control and move the quadcopter, independent variation in the speed of rotor can be changed accordingly. Change in speed of each rotor affects the total thrust, which can be further controlled to create desired total torque to navigate the quadcopter in all three directions. By giving the desired x-value and controlling the quadcopter to move on the given trajectory, pitch angle can be generated and controlled. Similarly, by controlling the movement along y direction of the craft, roll angle is achieved. To increase the pitch the thrust on motor 2 and motor 3 is decreased and for the diagonally placed motor 1 and motor 4, thrust is increased. This results in a moment, which tilts the quadcopter by creating a component of thrust in the x direction. On the same way, the thrust on motor 1 and motor 2 is decreased and for the diagonally placed motor 3 and motor 4, thrust is increased in a similar manner to maintain the position in the y direction.

Quadcopter Dynamics are derived and a mathematical formulation for the Quadcopter model is discussed in detail. Maintaining the altitude of the quadcopter, i.e., its position in the z-axis is achieved by apply equal amount of trust along each motors. Decreasing the thrust on one pair of motors arranged in one arm and increasing the thrust by an equal amount on the other two motors arranged on the other arm creates a moment about z-axis

of the quadcopter triggering it to yaw. To determine all the equation governing the motion of the quadcopter we need to analyse it in the inertial reference frame and the fixed body frame stated on chapter two. The position and velocity of the quadcopter is measured from inertial frame, whereas orientation and angular speeds are determined by body-fixed frame.

3.2 Reference frame alignment

The quad-copter has 6 degree of freedom; this means that six variables are needed to express its position and orientation in space. There are two reference frames essential to describe the behaviour of a quad-copter, the body frame and inertial frame.The reason why two frames are needed is that the quad-copter has many sensors, like gyroscope and accelerometer that give readings with respect to body frame to measure linear and angular velocity . Quadcopter also has other sensors, like GPS and magnetometer that give readings with respect to inertial frame to measure linear position and angular position. Therefore, a mean of transformation is required to derive system equations according to a single frame. The body frame is associated with the quad-copter and its origin corresponds to the centre of gravity (COG) of the vehicle. Since the quad-copter has a symmetrical shape with a central core and four identical rotors attached at its arms, the quad-copter COG is located at the core centre.

1. **Body-fixed reference frame** $(O_B, X_B, , Y_B, Z_B,)$
   - O_B coincides with center of the quad-copter.
   - Linear Velocity ($V^B = [U, V, W]^T$).
   - Angular Velocity ($W^B = [P, Q, R]^T$).
   - Force (F^B) and Torque (τ^B).
2. **Earth inertial reference frame**(O_E, X_E, Y_E, Z_E).
   - This frame is used to define the linear position $\left[\xi^E\right] = \left[X \quad Y \quad Z]\right]^T$ and the angular position $\left[\eta^E\right] = \left[\phi \quad \theta \quad \psi\right]^T$.
   - Represented by East-North-up coordinates.
   - The origin of the inertial frame is taken and fixed point on Earth.

Where ξ^E is determined by coordinates of the vector between O_B and O_E with respect to the inertial frame. η^E Is orientation of the body frame with respect to the inertial frame.

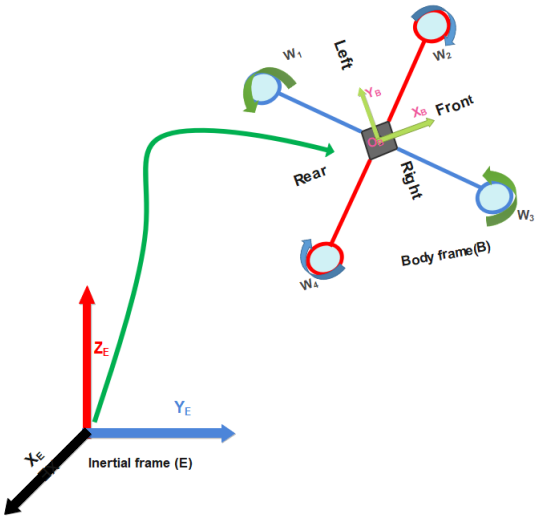


Figure (3.1) Inertial and Body Frames of References

3.3 Three- Dimensional rotation Description

3.3.1 Euler angle parametrization

Euler angles can be defined by a composition of rotations. The composition of elemental rotations about the axes of a certain coordinate system is sufficient to reach any target orientation. Using this principle, rotation is a rigid bogy in 3-dimensional space will be expressed using the composition of three consecutive rotations in the analysis to be followed. In flight mechanics, the Euler angles are often used the relation for transformations between those coordinate systems. These transformations are achieved with rotation matrices. This part will present the rotation matrices that transform between the two coordinate systems earth (E) and body frame (B). Consider that the quadcopter's roll, pitch and yaw angles are changed in relation to the earth fixed frame. ${}^{B}_{E}R$ The rotation matrix that transforms a linear quantity from earth fixed coordinates to body fixed coordinate. All three matrices representing the trigonometric equations are multiplied with in the sequence of Z-Y-X to get a single transformation matrix.

$$ {}^{B}_{E}R = R_Z R_Y R_X \tag{3.1} $$

$$ {}^{B}_{E}R = \begin{bmatrix} c\theta c\psi & c\theta s\phi & -s\theta \\ s\phi s\theta c\psi - c\phi s\psi & s\phi s\theta s\psi + c\phi c\theta & c\theta s\phi \\ c\phi s\theta c\psi + s\phi s\psi & s\phi s\theta c\phi - c\psi s\phi & c\phi c\theta \end{bmatrix} \tag{3.2} $$

Where$C\theta$ is an abbreviation of $cos(\theta)$ and $S(\theta)$ is an abbreviation of $sin(\theta)$.The same is true

for ψ and ϕ angles, ${}^{B}_{E}R$, inertial to body frame transformation.

Transformation matrix from the body reference frame to the inertial reference frame is obtained by transposing inertial to body frame transformation matrix.i.e ${}^{E}_{B}R = {}^{B}_{E}R^{T}$

3.3.2 Euler angular velocity and the body angular velocity conversion

The angular velocity $[p, q, r]$ changes over time, which a gyroscope sensor measured on-board of a quadcopter. So that the angular velocity about the body-fixed frame is not the same as the rate of change of the Euler angles $[\phi, \theta, \psi]$.To drive the relation among them is that based on the sequence of the rotation shown on figure below[6].

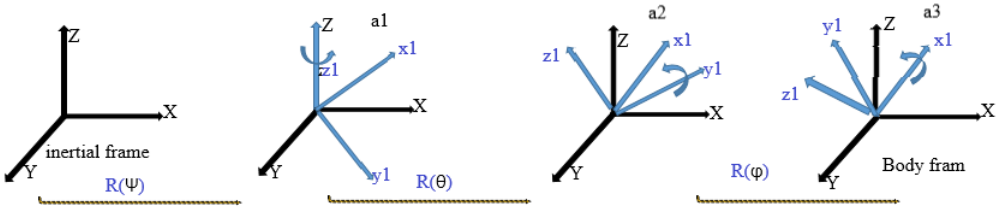


Figure (3.2) Sequence of Euler Angle Rotations.

The relationship between measured gyro rate and Euler angle rate expressed in scalar form multiplied by a scalar using right hand rule as:

$$px_1 + qy_1 + rz_1 = \dot{\psi}a_1 + \dot{\theta}a_2 + \dot{\phi}a_3$$

To write in matrix form.

$$
\begin{bmatrix} p \\ q \\ r \end{bmatrix} = R\phi R\theta \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix}^{a_1} + R\phi \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix}^{a_2} + \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix}^{a_3}
$$

$$
\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\psi & -sin\psi \\ 0 & sin\psi & cos\psi \end{bmatrix} \begin{bmatrix} cos\theta & 0 & -sin\theta \\ 0 & 1 & 0 \\ sin\theta & 0 & cos\theta \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\psi & -sin\psi \\ 0 & sin\psi & cos\psi \end{bmatrix} \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix}
$$

$$
\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & -s\theta \\ 0 & s\psi & s\phi c\theta \\ 0 & -sin\psi & c\psi c\theta \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{3.3}
$$

To find Euler rates from body angular velocity using matrix inverse rule:

$$
\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}_{inertial} = \begin{bmatrix} 1 & s\phi T\theta & c\phi T\theta \\ 0 & c\phi & -s\phi \\ 0 & -sin\phi sec\theta & c\phi sec\theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}_{bodyframe} \tag{3.4}
$$

Where (p,q,r)are roll rate pitch rate and yaw rate respectively.

3.4 Newton-Euler Model

In this section, the Newton-Euler method will be used to model the flight dynamics of the quadcopter. The equations of motion are developed as a combination of the translational and rotational parameters of the six-degree-of-freedom system, using Euler two laws of motion (newton second law and newton first law (linear momentum)). The representation of translational and rotational depends on the position vector, orientation vector, angular velocity and linear velocity. The behaviour of the mathematical model is then examined by simulating the flight of the quadcopter under different scenario.

The translational motion equations on the ground coordinate system by obtained by the method of Newton's second law. To represent inertial motion in the body coordinate frame , we must understand mathematical derivation of the equations of motion from newton second law of forces acting on quadcopter. Using the rules of calculus, we must use the chain rule of derivation to account for both the change due to the time derivative of the vector within the coordinate frame, as well as the time derivative of the coordinate frame's rotation.

Let the inertial set of unit vectors along the direction,$b = \begin{bmatrix} b_1 & b_2 & b_3 \end{bmatrix}$, which has an instantaneous angular velocity $\omega = \begin{bmatrix} p & q & r \end{bmatrix}$ we want to calculate derivative of linear velocity vector v with respect to body frame.

$$\begin{aligned} \Sigma F &= ma \\ F &= m\frac{dv}{dt} = m\frac{(d(u_{b1} + v_{b2} + w_{b3})}{dt} \end{aligned} \tag{3.5}$$

Where v is generalized linear velocity with respect to body frame. Using chain rule :

$$\begin{aligned} F &= m\frac{dv}{d_t} \\ &= m(\frac{d_u}{d_t}b_1 + \frac{db_1}{d_t}u + \frac{d_v}{dt}b_2 + \frac{db_2}{dt}v + \frac{dw}{dt}b_3 + \frac{db_3}{dt}w) \\ &= m(\frac{du}{dt}b_1 + \frac{dv}{dt}b_2 + \frac{dw}{dt}b_3) + m(\frac{db_1}{dt}u + \frac{db_2}{dt}v + \frac{db_3}{dt}w) \end{aligned} \tag{3.6}$$

$$\begin{aligned} F = ma &= m\frac{dv}{dt} \\ F_x &= m(\dot{u} - vr + wq) \\ F_y &= m(\dot{v} + wp - ur) \\ F_z &= m(\dot{w} + uq - vp) \end{aligned} \tag{3.7}$$

Rewrite in matrix form as follows.the detail derivation is derived in appendix A.2

$$F = \begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix} = \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & m \end{bmatrix} \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} + \begin{bmatrix} 0 & mw & -mv \\ -mw & 0 & mu \\ mv & -mu & 0 \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

From those two parts, the first part represent the change in inertial velocity within the body frame coordinate system. The second one represent velocity change due to the rotation of the coordinate frame. We can rewrite this equation as Coriolis' Theorem ,see appendix A.2

3.4.1 Momentum from Euler angle equation

Momentum $L = I * W$,where I = moment of inertia. Rotating torque on body frame is equal to rate of change of momentum at inertial frame . $\frac{dL}{dt}_{inertia} = \tau$. Using similar procedure to solve for the angular acceleration and rotational law of motion. Using Coriolis' Theorem

again for the time derivative of angular velocity, we can calculate the rotational motion as follows:

$\tau = L + wxL$.....euler equation .

$$\tau = \frac{d}{dt}\begin{bmatrix} I_x p & I_y q & I_z r \end{bmatrix}^T + \begin{bmatrix} p & q & r \end{bmatrix}^T \times \begin{bmatrix} I_x p & I_y q & I_z r \end{bmatrix}^T$$

$$\tau = \begin{bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} = \begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix} \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} + \begin{bmatrix} 0 & I_z r & -I_y q \\ -I_z r & 0 & I_x p \\ I_y q & -I_x p & 0 \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{3.8}$$

The generalized force vector can be represented by Λ can defined as

$$\Lambda = \begin{bmatrix} F^B \\ \tau^B \end{bmatrix} = \begin{bmatrix} F_x & F_y & F_z & \tau_x & \tau_y & \tau_z \end{bmatrix}^T$$

$$\Lambda = \begin{bmatrix} F_x \\ F_y \\ F_z \\ \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} = \begin{bmatrix} m & 0 & 0 & 0 & 0 & 0 \\ 0 & m & 0 & 0 & 0 & 0 \\ 0 & 0 & m & 0 & 0 & 0 \\ 0 & 0 & 0 & I_x & 0 & 0 \\ 0 & 0 & 0 & 0 & I_y & 0 \\ 0 & 0 & 0 & 0 & 0 & I_z \end{bmatrix} \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \\ \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & mw & -mv \\ 0 & 0 & 0 & -mw & 0 & mu \\ 0 & 0 & 0 & mv & -mu & 0 \\ 0 & 0 & 0 & 0 & I_z r & -I_y q \\ 0 & 0 & 0 & -I_z r & 0 & I_x p \\ 0 & 0 & 0 & I_y q & -I_x p & 0 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \\ p \\ q \\ r \end{bmatrix}$$

According to the nature of the quadrotor force contributions, Λ Can be divided in three parts.

$$\Sigma F = ma = F_g + F_{gro} + F_{propeller} \tag{3.9}$$

The first one is the force due to gravity, the second one is the gyroscopic effect produced by the rotation of each propeller and the third one is the force and torque directly produced by the main movement inputs.

3.4.2 Gravitational force in Body Frame

Gravity force always acts towards the center of the Earth, and is expressed in the inertial reference frame. Remember that z-inertial coordinate is directed in the upward direction, so gravity is a negative acceleration. However, all other forces are defined within the body

frame of the quadcopter. To include gravitational force in the dynamic equations of motion needs to convert it into body frame components as well, using Euler angle transformation matrix from inertial to body frame E^BR drive in the above. Let $F_g^E$and F_g^B represents the force of gravity in inertial and body frames of references respectively. To get force of gravity in body frame of reference, Force of gravity in inertial frame multiplied with ${}^B_E R$ inertial to body frame rotation matrix.

$$\begin{aligned} F_g^B &= {}^B_E R * F_g^E \\ F_g^E &= \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} \\ F_g^B &= {}^B_E R \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} \\ F_g^B &= \begin{bmatrix} mgsin\theta \\ -mgsin\phi cos\theta \\ -mgcos\theta cos\phi \\ 0 \\ 0 \\ 0 \end{bmatrix} \end{aligned} \tag{3.10}$$

3.4.3 Gyroscopic effects

Since two of the quadcopters propeller mounted on one arm are rotating clockwise (propeller 2 and 4) and the other two mounted on other arm rotating counter clockwise (propeller 1 and 3).An overall imbalance will happen when the algebraic sum of the rotor speeds is not equal to zero. This imbalance will cause a gyroscopic effect which is proportional to the roll

and pitch rates. The overall gyroscopic torque equation defines as:

$$F_{gro}^{B} = -\sum_{i=1}^{4} J_{TP}(w^{B}x \begin{bmatrix}0\\0\\1\end{bmatrix})(-1)^{k}\Omega_{k}$$

$$F_{gro}^{B} = J_{TP}\begin{bmatrix}0 & 0 & 0 & 0\\0 & 0 & 0 & 0\\0 & 0 & 0 & 0\\q & -q & q & -q\\p & -p & p & -p\\0 & 0 & 0 & 0\end{bmatrix}\Omega_{k} = J_{TP}\begin{bmatrix}0\\0\\0\\-q\\p\end{bmatrix}\Omega_{r} \tag{3.11}$$

$$\Omega_{k} = \begin{bmatrix}\Omega_{1} & \Omega_{2} & \Omega_{3} & \Omega_{4}\end{bmatrix}^{T}$$

$$\Omega_{r} = -\Omega_{1} + \Omega_{2} - \Omega_{3} + \Omega_{4}$$

Where

- J_{TP} is the total rotational moment of inertia around the propeller axis.
- Ω_r The overall propellers' speed.

3.4.4 Forces and torques produced by the principle motion inputs

The third contribution takes into account the forces and torques directly produced by way of the principle motion inputs. From aerodynamics consideration, it follows that both forces and torques are proportional to the squared propellers speed.

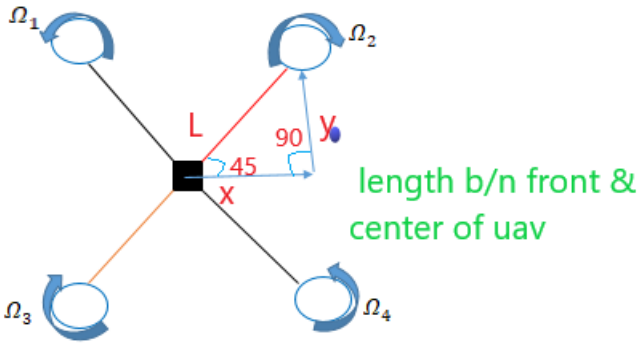


Figure (3.3) UAVs x-configuration for subsequent propellers distance.

Calculate the distance between center of quadrotor to center of middle point between subsequent propellers distance (x). Using sin law $X = (\sqrt{2}L)/2 = L/\sqrt{2}$

We assume that the control input of the four rotorcraft is $U = \begin{bmatrix} U_1 & U_2 & U_3 & U_4 \end{bmatrix}^T$. Where U_1 is the resultant force of lift force, (Altitude control input); U_2 is the resultant force that affects the roll angle of the aircraft; U_3 is the resultant force that affects the pitch angle of the vehicle; U_4 is the resultant force that affects the yaw angle of the quadcopter. The control input U is:

$$\begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ \Omega_r \end{bmatrix} = \begin{bmatrix} b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\ \frac{b_l}{\sqrt{2}}(\Omega_1^2 + \Omega_2^2 - \Omega_3^2 - \Omega_4^2) \\ \frac{b_l}{\sqrt{2}}(\Omega_1^2 - \Omega_2^2 - \Omega_3^2 + \Omega_4^2) \\ d(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2) \\ -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4 \end{bmatrix} \tag{3.12}$$

Where:-

- b indicates the thrust factor and d indicates the drift factor
- L= center of quadrotor to center of propeller distance
- Ω_i speeds for propeller i.

The translational motion of a four-rotor vehicle is achieved by the lift provided by four propellers, and since the propeller shaft of the aircraft is fixed, its lift direction is constant in the body coordinate system, i.e. perpendicular to the body. In the body coordinate system, the vertical force along the three-axis can be expanded as follows:

$$F^B = \begin{bmatrix} 0 \\ 0 \\ U_1 \end{bmatrix}$$

$$F_i = b\Omega_i^2$$

The rotational torque will drive Quadcopter in the air to do roll, pitch, yaw movement, resulting in changes in attitude. Specific roll, pitch, and yaw torques are expressed as

$$\tau_{propeller}^B = \begin{bmatrix} U_2 \\ U_3 \\ U_4 \end{bmatrix}$$

$$\Lambda = \begin{bmatrix} F^B \\ \tau^B \end{bmatrix}_{propeller} = \begin{bmatrix} 0 \\ 0 \\ U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} \tag{3.13}$$

$$\begin{aligned} \sum F &= m\dot{\Gamma} + w^B X \upsilon^B = F_g + F_{gro} + F_{propeller} \\ \upsilon &= \begin{bmatrix} u & v & w & p & q & r \end{bmatrix}^T \end{aligned} \tag{3.14}$$

$$\begin{cases} \dot{u} = (vr - wq) + gS\theta \\ \dot{v} = (wp - ur) - gC\theta S\phi \\ \dot{w} = (uq - vp) - gC\theta S\phi + \dfrac{U_1}{m} \\ \dot{p} = \dfrac{I_{yy} - I_{zz}}{I_{xx}} qr - \dfrac{J_{TP}}{I_{xx}} q\Omega + \dfrac{U_2}{I_{xx}} \\ \dot{q} = \dfrac{I_{zz} - I_{xx}}{I_{yy}} pr + \dfrac{J_{TP}}{I_{yy}} p\Omega + \dfrac{U_3}{I_{yy}} \\ \dot{r} = \dfrac{I_{xx} - I_{yy}}{I_{zz}} pq + \dfrac{U_4}{I_{zz}} \end{cases} \tag{3.15}$$

3.5 Hybrid system

The quadrotor dynamic system in the above is derived and written in the body fixed frame. This reference is widely used in six DOF rigid body equations. However, in this case it can be useful to explicit the dynamics of the system to a hybrid system composed of linear equations WRT inertial frame and angular equations WRT body frame. Therefore, the subsequent equations can be expressed inside the new "hybrid" body known as H-body. This new reference is derived because it is easy to express the dynamics combined with the control (in particular for the position in the earth inertial frame). To drive generalized quadrotor velocity vector WRT H-frame as follows. Forces and torques directly produced by the main movement inputs on inertial frame

$$F^E_{propeller} = {}^B_E R^T F^B_{propeller} = {}^B_E R^T \begin{bmatrix} 0 \\ 0 \\ U_1 \end{bmatrix}$$

$$\begin{bmatrix} F^E \\ \tau^B \end{bmatrix}_{propeller} = \begin{bmatrix} (C\phi S\theta C\psi + S\phi S\psi)\dfrac{u_1}{m} \\ (C\phi C\theta S\psi - S\phi C\psi)\dfrac{u_1}{m} \\ C\phi C\theta \dfrac{u_1}{m} \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} \tag{3.16}$$

Linear equation.....with respect to E-frame
Angular equation.....with respect to B-frame
Gravitational force affects only z- Direction of inertial frame

$$F^E_g = \begin{bmatrix} 0 \\ 0 \\ -mg \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Gyroscopic force is constant then use Newton's law $\sum F = ma = F_g + F_{gro} + F_{propeller}$

$$\begin{cases} \ddot{x} = (C\phi S\theta C\psi + S\phi S\psi)\dfrac{u_1}{m} \\ \ddot{y} = (C\phi S\theta S\psi - S\phi C\psi)\dfrac{u_1}{m} \\ \ddot{z} = C\phi C\theta\dfrac{u_1}{m} - g \\ \dot{p} = \dfrac{I_{yy} - I_{zz}}{I_{xx}}qr - \dfrac{J_{TP}}{I_{xx}}q\Omega + \dfrac{U_2}{I_{xx}} \\ \dot{q} = \dfrac{I_{zz} - I_{xx}}{I_{yy}}pr + \dfrac{J_{TP}}{I_{yy}}p\Omega + \dfrac{U_3}{I_{yy}} \\ \dot{r} = \dfrac{I_{xx} - I_{yy}}{I_{zz}}pq + \dfrac{U_4}{I_{zz}} \end{cases} \tag{3.17}$$

To calculate the propeller speeds from the control inputs by an inverse relationship formulated as:

$$\begin{cases} \Omega_1 = 0.5\sqrt{\dfrac{U_1}{b} + \dfrac{\sqrt{2}U_2}{bL} + \dfrac{\sqrt{2}U_3}{bL} + \dfrac{U_4}{d}} \\ \Omega_2 = 0.5\sqrt{\dfrac{U_1}{b} + \dfrac{\sqrt{2}U_2}{bL} - \dfrac{\sqrt{2}U_3}{bL} - \dfrac{U_4}{d}} \\ \Omega_3 = 0.5\sqrt{\dfrac{U_1}{b} - \dfrac{\sqrt{2}U_2}{bL} - \dfrac{\sqrt{2}U_3}{bL} + \dfrac{U_4}{d}} \\ \Omega_4 = 0.5\sqrt{\dfrac{U_1}{b} - \dfrac{\sqrt{2}U_2}{bL} + \dfrac{\sqrt{2}U_3}{bL} - \dfrac{U_4}{d}} \end{cases} \tag{3.18}$$

3.6 Model Verification

The Simulink model developed using non linear equations represents the quad-copter obeying real world limits. To verify the model by simply applying control signals of altitude and attitude inputs $(U1, U2, U3, U4)$ with the parameters of the quadrotor model listed on the table below. Simulink scope block is used for the visualization. Measure the propeller speeds based on the control inputs and analyse real dynamics of quadcopter with those speed results. Altitude (x, y, z) plotted in the same scope and attitude (ϕ, θ and ψ) are plotted in the same way for comparison and to observe the change in behaviour of the states of the quadcopter.

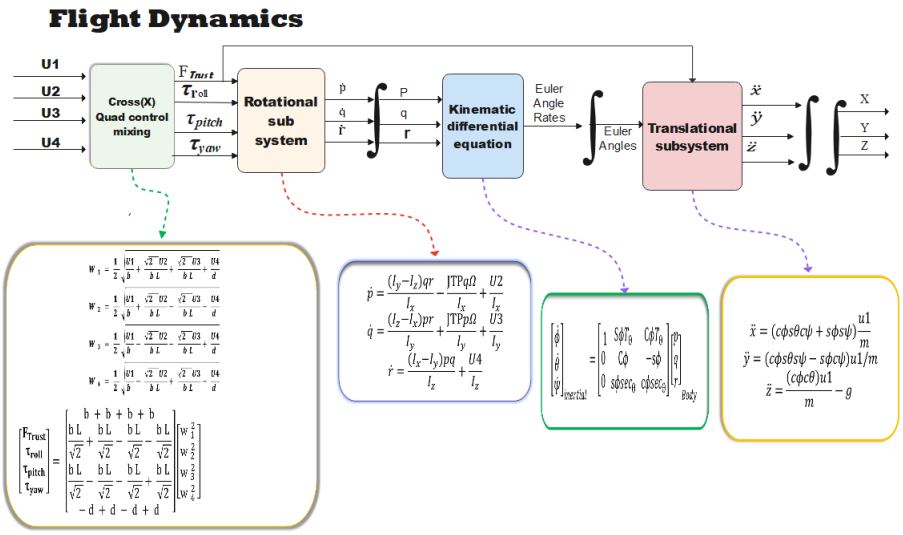


Figure (3.4) Overall mathematical model of quadcopter

Table (3.1) Parameters of the quad-copter [2]

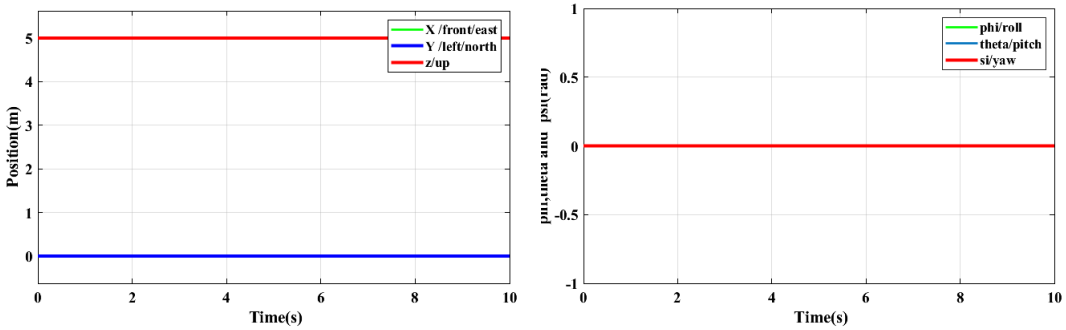
Parameter(unit)	Symbol	Value
Inertia around X axis(Nms^2)	I_x	0.0081
Inertia around Y axis(Nms^2)	I_y	0.0081
Inertia around Z axis(Nms^2)	I_z	0.0142
Gravitational acceleration(ms^{-2})	g	9.81
Drag coefficient(Nms^2)	d	1.1×10^{-6}
Trust coefficient(Ns^2)	b	54.2×10^{-6}
Mass of the quad-copter(Kg)	m	1
Inertia around propeller (Nms^2)	J_{TP}	104×10^{-6}
Arm length(m)	L	0.24

1. **Thrust** movement along z- axis, which is provide by applying altitude control signal U_1, which leads rotating all the motors at the same speed. Vehicle response is shown in below It can be seen that the altitude is changed while roll, pitch and yaw angles remain same. During this condition, we have three scenarios.

   - **Hovering state: -** if the value of trust $T = mg$, then the quadcopter stays on same height above the ground.

- **Fails downward state: -** The value of thrust $T < mg$, the vehicle fails down towards negative z- axis.
- **Upward movement:-** where the value of thrust $T > mg$, the vehicle goes up towards positive z- axis. In all case the speed of each propeller is equal i.e. $\Omega_1 = \Omega_2 = \Omega_3 = \Omega_4$.

The output curves of X, Y, Z when U_1=mg =9.81 and initial states of the quadcopter is at $5m$ above the ground on hovering state. Then the measured speed is$\Omega_1 = \Omega_2 = \Omega_3 = \Omega_4 = 212.2rad/s$. All angles stay on zero state.

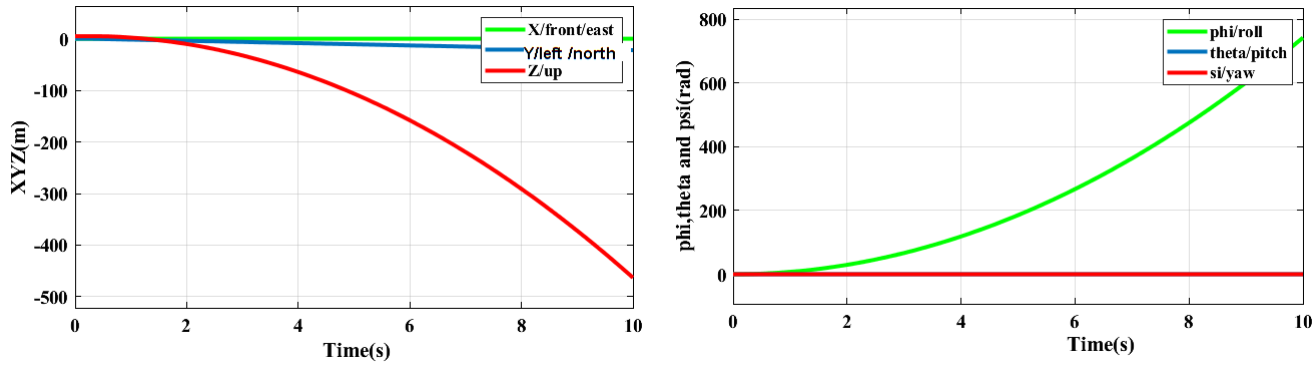


(a) Position at hovering state. (b) Angels at hovering state..

Figure (3.5) Model verification for hovering state output.

2. **Roll:-**movement of the quadcopter towards Y-axis and Z-axis, When we apply a control signal $U_2 = 0.1$, and $U_1 = 9.81$,the corresponding position and orientation is shown below.

   - Which leads the speed $\Omega_1 = \Omega_2 = 219rad/s$ and $\Omega_3 = \Omega_4 = 206.2rad/s$ will cause roll angle change.
   - The quadcopter moves down ward along negative z-axis and towards negative y-axis (south) and x-axis remains zero.

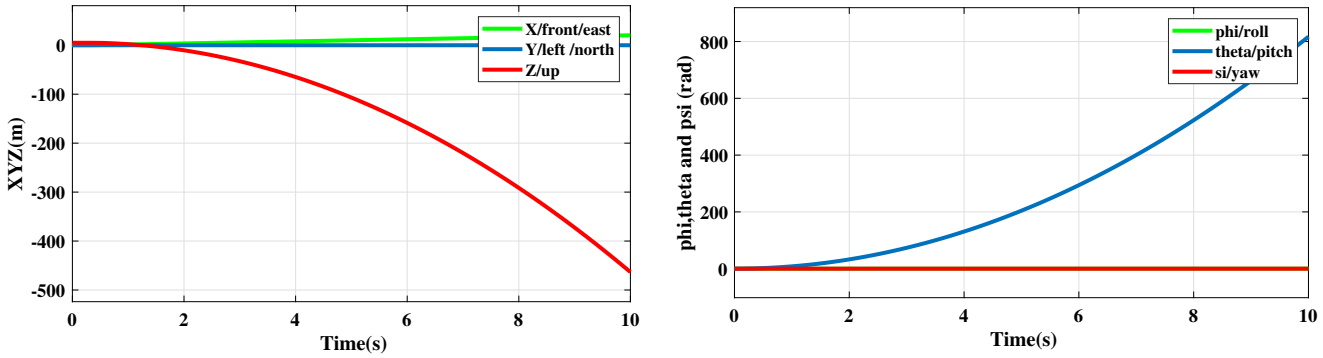


(a) Roll movement positions. (b) Roll movement orientations.

Figure (3.6) Roll movement Model verification.

3. **Pitch:-**movement of the quadcopter towards $Y-$axis and $Z-$axis, When we apply $U_3 = 0.11$, and $U_1 = 9.81$,the corresponding position and orientation is shown below.

   - Which leads an increasing the speed $\Omega_1 = \Omega_4 = 219 rad/s$ and decreasing $\Omega_2 = \Omega_3 = 206.2 rad/s$ speeds at the same time will cause pitch angle change.
   - The quadcopter moves down ward along negative z-axis and towards positive x-axis (east) and y-axis remains zero and will cause the change of Pitch angle.



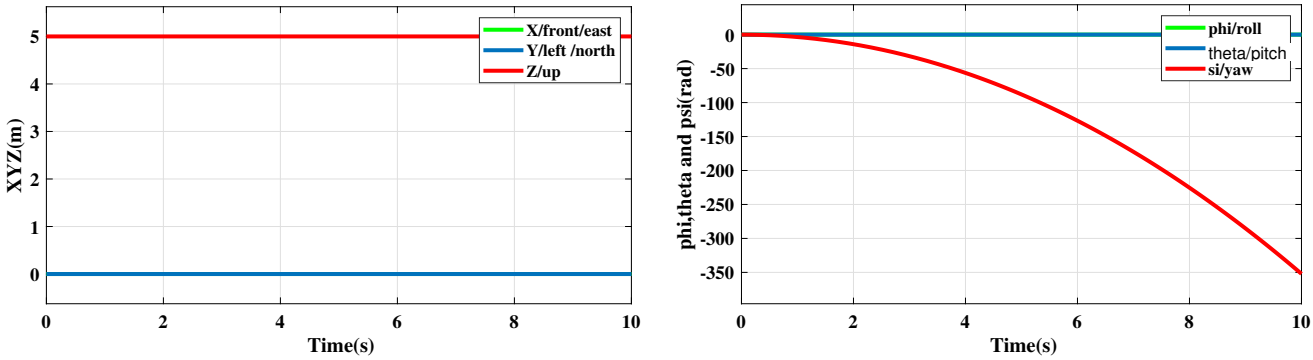
(a) Pitch movement positions. (b) Pitch movement orientations.

Figure (3.7) Pitch movement Model verification.

4. **Yaw:-** When we apply $U_3 = 0.1$, and $U_1 = 9.81$,the corresponding position and orientation is shown below(the green is X, blue is Y and red is Z)

   - Which leads an increasing the speed of $\Omega_1 = \Omega_3 = 260.7 rad/s$ and decreasing the speed of $\Omega_2 = \Omega_4 = 150.1 rad/s$.

- The positions are not affected by yaw movement The quadcopter rotates CW or CCW based on the command.



(a) yaw movement positions.

(b) Yaw movement orientations.

Figure (3.8) Yaw movement Model verification.

All the above output verifies that the model is built correctly and works properly.

Chapter 4

Controller design

The control engineer researchers always aim to design controllers that are insensitive to system uncertainties, external disturbances and un-modelled dynamics for a long time. When all quadcopters industry are considered, there are changes in system performance due to various reasons such as propellers effect, mechanical stresses, mechanical fatigue, expansion and friction, external disturbances like, measurement errors, environmental temperature changes, wind speed and direction and rain affect system performance. Such parameter changes, external disturbances and modelling uncertainties are factors that reduce system performance. The theory of control systems is a deterministic control strategy used to control these dynamic systems. Fractional order sliding mode control is considered as a subclass of the variable structure systems theory. FOSMC is a deterministic, nonlinear, and robust control method switches on a specified sliding surface by driving the phase trajectory to it. This is owing to its simplicity of usage as well as its sensitivity to restricted external disturbances and parameter uncertainty .

4.1 Fractional-order calculus (FOC)

Fractional analytics is a field of mathematics that studies the potential of obtaining real or complex number powers of the differentiation operator and generalizes a function's derivative or integral to non-integer order. Fractional-order calculation (FOC) is a particularly appealing way for achieving more sensitive controller. The fractional order calculus is a generalization of the calculus of first order. represented with ${}_{\alpha}D_{t}^{\beta}f(t)$. The FOC integral

and differentiator are expressed as [22]:-

$$ {}_{\alpha}D_{t}^{\beta}f(t) = \begin{cases} \dfrac{d^{\beta}}{dt^{\beta}} & R(\beta) > 0 \\ 1 & R(\beta) = 0 \\ \displaystyle\int_{\alpha}^{t} (d\tau)^{-\beta} & R(\beta) < 0 \end{cases} \tag{4.1} $$

Where D represents the fractional calculus operator, β represents the fractional order of a real or complex number, α and t are the limits of the operation,$R(\beta)$ is the real part of β. The most common fractional operators considered in different literatures are the Riemann–Liouville (RL) definition, the Caputo fractional derivatives (CFD) and Grunwald -Letnikov (GL) definition formulated as follows.

- The Riemann-Liouville (RL) definition

$$ {}_{\alpha}D_{t}^{\beta}f(t) = \frac{1}{\Gamma(n-\beta)}\frac{d^{n}}{dt^{n}}\int_{\alpha}^{t}\frac{f(\tau)}{(t-\tau)^{\beta-n+1}}d\tau \tag{4.2} $$

- The Caputo's definition

$$ {}_{\alpha}D_{t}^{\beta}f(t) = \frac{1}{\Gamma(n-\beta)}\int_{\alpha}^{t}\frac{f^{n}(\tau)}{(t-\tau)^{\beta-n+1}}d\tau \tag{4.3} $$

  In these definitions of fractional order , $n-1 < \alpha < n$,where n is minimum integer greater than α and ,$\Gamma(.)$ is the widely known Euler's gamma function whose definition is:

$$ \Gamma(x) = \int_{0}^{\infty} e^{-t}t^{x-1}dt \quad \text{for } t > t_0 $$

In simulation studies, download a fractional-order modeling and control toolbox called FOMCON and add in Matlab software for fractional-order calculus.

4.2 Fractional order Sliding Mode Controller (FOSMC)

The most significant aspect of FOSMC is that it ensures the state's sliding motion on the sliding surface. As a result, the movement of the controlled system's states can be divided into the reaching phase and the sliding phase.

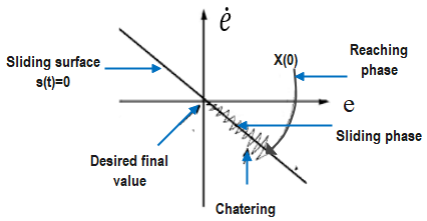


Figure (4.1) Sliding surface of FOSMC

- **The first phase (reaching phase)**, it is a non-robust phase where the system is driven to the sliding surface from any initial point. The discontinuous controller of equation 4.4 responsible for this task. Increasing the FOSMC's discontinuous control gain k will reduce the reaching phase, but this will cause considerable chattering. Chattering occurs when the control signal oscillates at a high frequency with a finite amplitude, which damages mechanical system. To address this issue, a variety of approaches has been offered, like use of a quasi function to approximate the signum function.

$$U_{Ds}(t) = ksign(s(t)) \tag{4.4}$$

  Chattering problem can be reduced by using quasi function below,where σ is small positive number.

$$U_{Ds}(t) = \frac{ks(t)}{|s(t)| + \sigma} \tag{4.5}$$

- **Second phase (sliding phase)**, The trajectories are unaffected by disturbances, parameter variation and un-modelled dynamics in this phase. FOSMC is a robust control mechanism because of this property. A long reaching time happens when a system's initial value is far from the sliding surface. As a result, control performance suffers significantly, and robustness cannot be guaranteed throughout the reaching phase. Due to those, sliding phase is introduce. The equivalent control laws have the responsibilities to perform those tasks and obtained by setting derivative of sliding surface equate to zero $\dot{s} = 0$.

- In this phase the system slides on the sliding surface (decision rule).
- System dynamics is independent of system equation.
- The whole system is driven by sliding surface.
- Order of the system equation is reduced.

The total control law is equal to the sum of both equivalent controller and discontinuous controller.

$$U = U_{equ}(t) + U_{Ds}(t) \tag{4.6}$$

Based on the simulation results in the previous section, we can get the influence of the control signal and motor speed on the attitude angle and altitude of the quad-copter. It can be deduced that if we want to get the corresponding attitude angle, we can get the change process of the motor. For the six degrees of freedom of a four-rotor vehicle, to obtain straightforward and robust control structure , FOSMC algorithm is used in order to solve this problem. The control approach divided into two loops . The inner control loop, which is the attitude, roll angle, pitch angle, yaw angle. The other one is outer control loop, which is the position, or displacement, including the upper and lower, front and rear, left and right direction displacement. There is a consideration how to control the front-to-back, left right positional relationship of the quad-copter for the effect of the motor speed on the attitude angle. Therefore, the control system to establish position control and attitude angle closed-loop control. The control system input is the target positions X_r,Y_r and Z_r, and the target yaw angle, ψ_r, cause the aircraft to produce the corresponding movements, and consequently the corresponding displacements. The overall control system block diagram is shown in figure below.

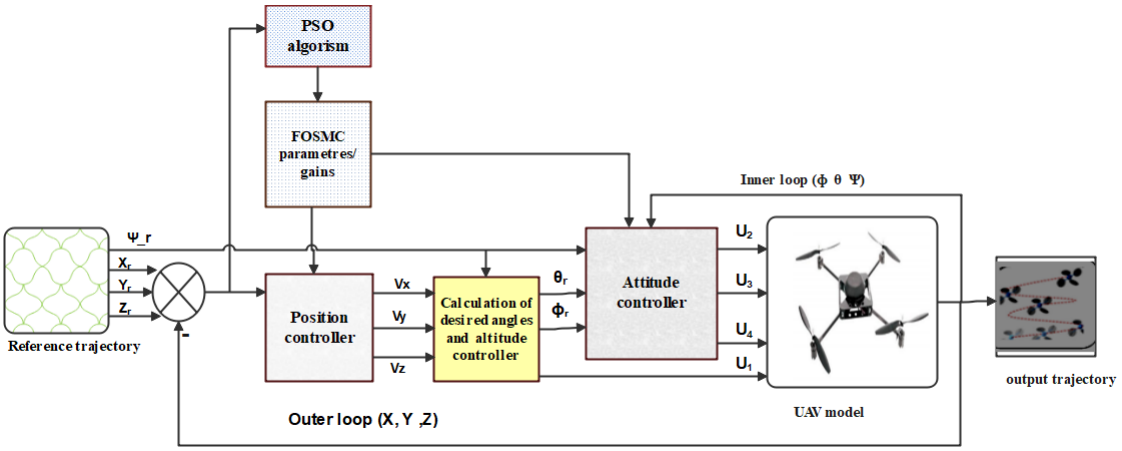


Figure (4.2) Control scheme block diagram

4.2.1 Outer Control Loop

Outer control loop is used because the quadcopter is under actuated system and it is impossible to control all of the six DOF quadcopter directly. The inner loop directly controls

$4DOF$, three angles roll, pitch and yaw and altitude. To be able to control X_E and Y_E positions indirectly, outer loop is used. Outer control loop controller outputs V_x, V_y, and V_z are used to determine the desired roll angle ϕ_r and pitch angle θ_r.
Let us define the position error and its derivative as:-

$$\begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix} = \begin{bmatrix} X_r - X \\ Y_r - Y \\ Z_r - Z \end{bmatrix}, \begin{bmatrix} \dot{e_x} \\ \dot{e_y} \\ \dot{e_z} \end{bmatrix} = \begin{bmatrix} \dot{X}_r - \dot{X} \\ \dot{Y}_r - \dot{Y} \\ \dot{Z}_r - \dot{Z} \end{bmatrix}, \begin{bmatrix} \ddot{e_x} \\ \ddot{e_y} \\ \ddot{e_z} \end{bmatrix} = \begin{bmatrix} \ddot{X}_r - \ddot{X} \\ \ddot{Y}_r - \ddot{Y} \\ \ddot{Z}_r - \ddot{Z} \end{bmatrix} \tag{4.7}$$

Then, a sliding surface should be determined. For this, the order of switching functions should be less than the order of the plant model. To implement FOSMC to quadcopter dynamics, the sliding surfaces X, Y, Z directions are defined respectively as.

$$\begin{cases} s_x = c_x e_x + D^{\beta_x} \dot{e_x} = c_x e_x + D^{\beta_x + 1} e_x \\ s_y = c_y e_y + D^{\beta_y} \dot{e_y} = c_y e_y + D^{\beta_y + 1} e_y \\ s_z = c_z e_z + D^{\beta_z} \dot{e_z} = c_z e_z + D^{\beta_z + 1} e_z \end{cases} \tag{4.8}$$

After choosing the sliding surface, control law can be calculated . This control law has to converge the system to the sliding surface. When it reaches the sliding surface, it must remain on the surface. Design Fractional order sliding mode controller for position tracking as follows:- From the equivalent control law, $\dot{s_i} = 0$ where $i = x, y, z$.

$$\begin{aligned} \dot{s_x} &= c_x \dot{e_x} + D_x^{\beta} \ddot{e_x} \\ &= c_x \dot{e_x} + D^{\beta_x + 2} e_x = 0 \\ &= c_x e_x + D_x^{\beta} (\ddot{X}_r - \ddot{X}) = 0 \end{aligned} \tag{4.9}$$

From virtual control law substitute $\ddot{X} = V_x$

$$0 = c_x \dot{e_x} + D_x^{\beta} (\ddot{X}_r - V_{xequ}) \tag{4.10}$$

$$V_{xequ} = c_x D^{-\beta_x + 1} e_x + \ddot{X}_r \tag{4.11}$$

Then the total virtual controller law along x-axis is the sum of the continuous and discontinuous control signal stated above as:-

$$\begin{aligned} V_x &= V_{xequ} + V_{xDs} \\ V_x &= c_x D^{-\beta_x + 1} e_x + \ddot{X}_r) + k_x \frac{s(x)}{|s(x)| + \sigma} \end{aligned} \tag{4.12}$$

Where, c_x and k_x are the equivalent and discontinuous control gains respectively. These control gains are selected through Particle Swarm Optimization detailed in bellow given a finite range, The lower of the discontinuous gains k_x is determined from the maximum system uncertainty to satisfy Lyapunov stability. and β_x fractional order parameters between $(0, 1)$.

Condition on the switching gain $(k_i, i = x, y, z)$

The Lyapunov function is a positive scalar function of the system state variables $V(x) > 0$. The control law must satisfy rate of Lyapunov function decrease as time increase, i.e $\dot{V}(x) < 0$. The idea is to choose a scalar function $s(x)$ to guarantee that the attraction of the variable to be controlled to its reference value and to design a controller signal or command u such that the square of the surface corresponds to a Lyapunov function. The Lyapunov function is defined as $V_x = \frac{1}{2}s_x^2$. To get the minimum value of the discontinuous controller gains let as proof:

$$\begin{aligned}
\dot{V_x} &= s_x \dot{s_x} < 0 \\
&= s_x(c_x e_x + D_x^{\beta}(\ddot{X_r} - \ddot{X})) \\
&= s_x(c_x e_x + D_x^{\beta}(\ddot{X_r} - V_{1equ})) \\
&= s_x(c_x e_x + D_x^{\beta}(\ddot{X_r} - (c_x D^{-\beta_x+1} e_x + \ddot{X_r}) + k_x sign(x)))) \\
&= s_x(c_x e_x + D_x^{\beta}\ddot{X_r} - c_x \dot{e_x} - D^{\beta_x}\ddot{X_r}) - D^{\beta_x} k_x sign(x)) <= 0 \\
&= s_x(-k_x sign(x)) <= 0 \\
&= -k_x |s_x| <= 0
\end{aligned} \tag{4.13}$$

To make The Lyapunov function of equation 4.13 is valid, k_x must be greater than zero.

To calculate the virtual controls along y-axis and z- axis apply the same procedure to x-axis to drive the error approaches to zero . From the equivalent control law, $\dot{s_y} = 0$.

$$\begin{aligned}
\dot{s_y} &= c_y \dot{e_y} + D_y^{\beta} \ddot{e_y} \\
&= c_y \dot{e_y} + D^{\beta_y+2} e_y = 0 \\
&= c_y e_y + D_y^{\beta}(\ddot{Y_r} - \ddot{Y}) = 0
\end{aligned} \tag{4.14}$$

From virtual control law substitute $\ddot{Y} = V_y$

$$c_y \dot{e_y} + D_y^{\beta}(\ddot{Y_r} - V_{yequ}) = 0 V_{yequ} = c_y D^{-\beta_y+1} e_y + \ddot{Y_r} \tag{4.15}$$

The virtual controller along Y-axis is the sum of the continuous and discontinuous control

signal.

$$\begin{aligned} V_y &= V_{yequ} + V_{yDs} \\ V_y &= c_y D^{-\beta_y+1} e_y + \ddot{Y}_r) + k_y \frac{s(y)}{|s(y)| + \sigma} \end{aligned} \tag{4.16}$$

$$\begin{aligned} \dot{s_z} &= c_z \dot{e_z} + D_z^{\beta} \ddot{e_z} \\ &= c_z \dot{e_z} + D^{\beta_z+2} e_z = 0 \\ &= c_z e_z + D_z^{\beta} (\ddot{Z}_r - \ddot{Z}) = 0 \end{aligned} \tag{4.17}$$

From virtual control law substitute $\ddot{Z} = V_z$

$$c_z \dot{e_z} + D_z^{\beta} (\ddot{Z}_r - V_{zequ}) = 0 V_{zequ} = c_z D^{-\beta_z+1} e_z + \ddot{Z}_r \tag{4.18}$$

Total virtual controller along z-axis.

$$\begin{aligned} V_z &= V_{zequ} + V_{zDs} \\ V_z &= c_z D^{-\beta_z+1} e_z + \ddot{Z}_r) + k_z \frac{s(z)}{|s(z)| + \sigma} \end{aligned} \tag{4.19}$$

The range of the discontinuous controller gains along y-axis and z-axis is proved by the same method to x-axis. The outer loop controller (position) controller of the quadcopter to track the given trajectory along x-axis, y-axis and z-axis in the $3 - D$ space. For under-actuated part, ϕ, θ, U_1 are used to create three virtual control inputs to control the output separately. Physically, these virtual controls mean that the motion along x, y, and z are controlled indirectly by the three common inputs (ϕ, θ, U_1). These virtual controls are given by Equation 4.20:

$$\begin{cases} V_x = c_x D^{-\beta_x+1} e_x + \ddot{X}_r) + k_x \dfrac{s(x)}{|s(x)| + \sigma} \\ V_y = c_y D^{-\beta_y+1} e_y + \ddot{Y}_r) + k_y \dfrac{s(y)}{|s(y)| + \sigma} \\ V_z = c_z D^{-\beta_z+1} e_z + \ddot{Z}_r) + k_z \dfrac{s(z)}{|s(z)| + \sigma} \end{cases} \tag{4.20}$$

Those virtual inputs V_x, V_y, and V_z written in equation 4.20, which are used to determine the desired roll ϕ_r and pitch θ_r and attitude controller U_1 . Notice now that, when θ and ϕ are managed to approach ϕ_r and θ_r as soon as possible, the FOSMC strategy discussed above allows to reach the sliding surface $s_x = 0$, $s_y = 0$ and $s_z = 0$ where the tracking errors e_x , e_y and e_z tend to zero exponentially. In this paper, FOSMC action is implemented to

allow that $\theta \to \theta_r$, $\phi \to \phi_r$ and $\psi \to \psi_r$, by inner control loop explained below. After a simple calculation, the lift force/altitude control signal and the desired Euler angles written as:

$$\begin{cases} U_1 = m * \sqrt{V_x^2 + V_y^2 + (V_x + g)^2} \\ \theta_r = arctan(\dfrac{c\psi_r * V_x + s\psi_r * V_y}{V_z + g}) \\ \psi_r = arctan(\dfrac{c\theta_r(s\psi_r * V_x - c\psi_r * V_y}{V_z + g}) \end{cases} \tag{4.21}$$

The derivation of those desired Euler angles and altitude controller is is described in appendix A.3

4.2.2 Inner Control Loop

Inner control loop controls quadcopter altitude and attitude. Desired altitude and yaw angle are given by the task that quadcopter needs to accomplish and desired roll and pitch angles, calculated from the outer control loop. Sensors give measured altitude, roll angle, pitch angle and yaw angle. Inner loop outputs are three control variables (U_2, U_3, U_4). To design FOSMC stabilizes the tracking errors of the attitude subsystem by generating the input control signals(U_2, U_3, U_4). Define the tracking errors and its derivatives of Euler angles of the inner loop as

$$\begin{bmatrix} e_\phi \\ e_\theta \\ e_\psi \end{bmatrix} = \begin{bmatrix} \phi_r - \phi \\ \theta_r - \theta \\ \psi_r - \psi \end{bmatrix}, \begin{bmatrix} \dot{e_\phi} \\ \dot{e_\theta} \\ \dot{e_\psi} \end{bmatrix} = \begin{bmatrix} \dot{\phi_r} - \dot{\phi} \\ \dot{\theta_r} - \dot{\theta} \\ \dot{\psi_r} - \dot{\psi} \end{bmatrix}, \begin{bmatrix} \ddot{e_\phi} \\ \ddot{e_\theta} \\ \ddot{e_\psi} \end{bmatrix} = \begin{bmatrix} \ddot{\phi_r} - \ddot{\phi} \\ \ddot{\theta_r} - \ddot{\theta} \\ \ddot{\psi_r} - \ddot{\psi} \end{bmatrix} \tag{4.22}$$

In the same procedure design of the outer loop control, the sliding surfaces for roll, pitch and yaw angles and its derivatives can be written as:

$$\begin{cases} s_\phi = c_\phi e_\phi + D^{\beta_\phi} \dot{e_\phi} = c_\phi e_\phi + D^{\beta_\phi + 1} e_\phi \\ s_\theta = c_\theta e_\theta + D^{\beta_\theta} \dot{e_\theta} = c_\theta e_\theta + D^{\beta_\theta + 1} e_\theta \\ s_\psi = c_\psi e_\psi + D^{\beta_\psi} \dot{e_\psi} = c_\psi e_\psi + D^{\beta_\psi + 1} e_\psi \end{cases} \tag{4.23}$$

- **Design FOSMC for the Roll moment control variable** U_2

$$
\begin{aligned}
\dot{s_\phi} &= c_\phi \dot{e_\phi} + D_\phi^\beta \ddot{e_\phi} = 0 \\
&= c_\phi \dot{e_\phi} + D_\phi^\beta (\ddot{\phi_r} - \ddot{\phi}) = 0 \\
&= c_\phi \dot{e_\phi} + D_\phi^\beta (\ddot{\phi_r} - (\frac{I_y - I_z}{I_x}\dot{\theta}\dot{\psi} - \frac{J_{TP}}{I_x}\dot{\theta}\Omega + \frac{U_{2equ}}{I_x} = 0 \\
D_\phi^\beta U_{2equ} &= (c_\phi \dot{e_\phi} + D_\phi^\beta \ddot{\phi_r} - D_\phi^\beta \frac{I_y - I_z}{I_x}\dot{\theta}\dot{\psi} + D_\phi^\beta \frac{J_{TP}}{I_x}\dot{\theta}\Omega) I_x \\
U_{2equ} &= (c_\phi D^{-\beta_\phi} \dot{e_\phi} + \ddot{\phi_r} - (\frac{I_y - I_z}{I_x}\dot{\theta}\dot{\psi}) + (\frac{J_{TP}}{I_x}\dot{\theta}\Omega)) I_x \\
U_{2Ds} &= k_\phi \frac{s(\phi)}{|s(\phi)| + \sigma} \\
U_2 &= U_{2equ} + U_{2ds} \\
U_2 &= (c_\phi D^{-\beta_\phi} \dot{e_\phi} + \ddot{\phi_r} - (\frac{I_y - I_z}{I_x}\dot{\theta}\dot{\psi}) + (\frac{J_{TP}}{I_x}\dot{\theta}\Omega)) I_x + k_\phi \frac{s(\phi)}{|s(\phi)| + \sigma}
\end{aligned}
\tag{4.24}
$$

Where k_ϕ and c_ϕ are roll angle controller parameters and β_ϕ is fractional number $0 < \beta_\phi < 1$, ϕ_r is the desired/reference roll angle and ϕ is the measured roll angle. In the same way as position control to find the ranges of the discontinuous controller gain using lyaponuve stability theorem:

$$
\begin{aligned}
V_\phi &= 1/2 s_\phi^2 \\
\dot{V_\phi} &= s_\phi \dot{s_\phi} <= 0 \\
&= s_\phi (c_\phi \dot{e_\phi} + D^{\beta_\phi} (\ddot{\phi_r} - \ddot{\phi})) <= 0 \\
&= s_\phi (c_\phi \dot{e_\phi} + D^{\beta_\phi} (\ddot{\phi_r} - (\frac{I_y - I_z}{I_x})\dot{\theta}\dot{\psi} - \frac{J_{TP}}{I_x}\dot{\theta}\Omega - \frac{U_2}{I_x})) <= 0 \\
&= s_\phi (-D^{\beta_\phi} k_\phi sign(s_\phi) I_x) <= 0 \\
&= D_\phi^\beta - k_\phi \frac{|s_x|}{I_\phi} <= 0 \\
&= -k_\phi |s_x| I_\phi <= 0
\end{aligned}
\tag{4.25}
$$

- **Design FOSMC for the pitch moment control variable** U_3

$$
\begin{aligned}
\dot{s_\theta} &= c_\theta \dot{e_\theta} + D_\theta^\beta \ddot{e_\theta} = 0 \\
&= c_\theta \dot{e_\theta} + D_\theta^\beta (\ddot{\theta_r} - \ddot{\theta}) = 0 \\
&= c_\theta \dot{e_\theta} + D_\theta^\beta (\ddot{\theta_r} - (\frac{I_z - I_x}{I_y}\dot{\phi}\dot{\psi} + \frac{J_{TP}}{I_y}\dot{\phi}\Omega + \frac{U_{3equ}}{I_y})) = 0 \\
D_\theta^\beta U_{3equ} &= (c_\theta \dot{e_\theta} + D_\theta^\beta \ddot{\theta_r} - D_\theta^\beta (\frac{I_z - I_x}{I_y}\dot{\phi}\dot{\psi}) - D_\theta^\beta (\frac{J_{TP}}{I_y}\dot{\theta}\Omega))I_y \\
U_{3equ} &= (c_\theta D^{-\beta_\theta} \dot{e_\theta} + \ddot{\theta_r} - (\frac{I_z - I_x}{I_y}\dot{\phi}\dot{\psi}) - (\frac{J_{TP}}{I_y}\dot{\phi}\Omega))I_y \\
U_{3Ds} &= k_\theta \frac{s(\theta)}{|s(\theta)| + \sigma} \\
U_3 &= U_{3equ} + U_{3ds} \\
U_3 &= (c_\theta D^{-\beta_\theta} \dot{e_\theta} + \ddot{\theta_r} - (\frac{I_z - I_x}{I_y}\dot{\phi}\dot{\psi}) - (\frac{J_{TP}}{I_y}\dot{\phi}\Omega))I_y + k_\theta \frac{s(\theta)}{|s(\theta)| + \sigma}
\end{aligned}
\tag{4.26}
$$

Similar to the roll control, k_θ, β_θ and c_θ are three pitch angle controller parameters. e_θ is the pitch angle error . θ_r is the desired/reference pitch angle and θ is the measured pitch angle.

- **Design FOSMC for the yaw moment control variable** U_4

$$
\begin{aligned}
\dot{s_\psi} &= c_\psi \dot{e_\psi} + D_\psi^\beta \ddot{e_\psi} = 0 \\
&= c_\psi \dot{e_\psi} + D_\psi^\beta (\ddot{\psi_r} - \ddot{\psi}) = 0 \\
&= c_\psi \dot{e_\psi} + D_\psi^\beta (\ddot{\psi_r} - (\frac{I_x - I_y}{I_z})\dot{\theta}\dot{\phi} + \frac{U_{4equ}}{I_z}) = 0 \\
D_\psi^\beta U_{4equ} &= c_\psi \dot{e_\psi} + D_\psi^\beta \ddot{\psi_r} - D_\psi^\beta (\frac{I_x - I_y}{I_z}\dot{\theta}\dot{\psi}) \\
U_{4equ} &= (c_\psi D^{-\beta_\psi} \dot{e_\psi} + \ddot{\psi_r} - (\frac{I_x - I_y}{I_z})\dot{\theta}\dot{\phi})I_y \\
U_{4Ds} &= k_\psi \frac{s(\psi)}{|s(\psi)| + \sigma} \\
U_4 &= U_{4equ} + U_{4ds} \\
U_4 &= (c_\psi D^{-\beta_\psi} \dot{e_\psi} + \ddot{\psi_r} - (\frac{I_x - I_y}{I_z})\dot{\theta}\dot{\phi})I_y + k_\psi \frac{s(\psi)}{|s(\psi)| + \sigma}
\end{aligned}
\tag{4.27}
$$

Where $k_{\psi,\psi}$ and c_ψ are three yaw angle controller parameters. e_ψ is the yaw angle error, and ψ_r is the desired/reference yaw angle and ψ is the measured yaw angle

The position and attitude FOSM controllers of the quadcopter is summarized as:

$$\begin{cases} V_x = c_x D^{-\beta_x+1} e_x + \ddot{X_r}) + k_x \dfrac{s(x)}{|s(x)| + \sigma} \\ V_y = c_y D^{-\beta_y+1} e_y + \ddot{Y_r}) + k_y \dfrac{s(y)}{|s(y)| + \sigma} \\ V_z = c_z D^{-\beta_z+1} e_z + \ddot{Z_r}) + k_z \dfrac{s(z)}{|s(z)| + \sigma} \\ U_1 = m * \sqrt{V_x^2 + V_y^2 + (V_x + g)^2} \\ U_2 = (c_\phi D^{-\beta_\phi} \dot{e_\phi} + \ddot{\phi_r} - (\dfrac{I_y - I_z}{I_x} \dot{\theta}\dot{\psi}) + (\dfrac{J_{TP}}{I_x} \dot{\theta}\Omega))I_x + k_\phi \dfrac{s(\phi)}{|s(\phi)| + \sigma} \\ U_3 = (c_\theta D^{-\beta_\theta} \dot{e_\theta} + \ddot{\theta_r} - (\dfrac{I_z - I_x}{I_y} \dot{\phi}\dot{\psi}) - (\dfrac{J_{TP}}{I_y} \dot{\phi}\Omega))I_y + k_\theta \dfrac{s(\theta)}{|s(\theta)| + \sigma} \\ U_4 = (c_\psi D^{-\beta_\psi} \dot{e_\psi} + \ddot{\psi_r} - (\dfrac{I_x - I_y}{I_z} \dot{\theta}\dot{\phi})I_y + k_\psi \dfrac{s(\psi)}{|s(\psi)| + \sigma} \end{cases} \tag{4.28}$$

4.3 FOSMC Gain Tuning

The $FOSMC$ position control gains such as, $\begin{bmatrix} k_x & \beta_x & c_x & k_y & \beta_y & c_y & k_z & c_z & \beta_z \end{bmatrix}$ and orientation control gains $\begin{bmatrix} k_\phi & \beta_\phi & c_\phi & k_\theta & \beta_\theta & c_\theta & k_\psi & \beta_\psi & c_\psi \end{bmatrix}$ in the quadcopter control laws of equation 4.28 are tuned using PSO. A concept for the optimization of non-linear system using particle swarm method is introduced by[23] and developed further in [24], was inspired by the movements of flocks of birds and schools of fish and has been found to be highly effective in solving complicated optimization problems. PSO is an evolutionary algorithm under stochastic iterative optimization methods , which has some notable advantages over those and other optimization techniques. initially PSO does not require any a priori information about the search space or variables. It is, therefore, capable of executing a multi-variable optimization of highly complicated systems with variables that may not be directly or evidently related in a hyper-spatial search region. PSO works to minimize objective function by comparing multiple samples of the optimization variables in successive steps. The values of the optimization variables for the next step are then subjective to tend toward a set, which gives the lowest fitness. This is evaluated with any constant number of samples (or particles) for any number of steps until the loop is terminated. Moreover, PSO is relatively simple when compared with other optimization techniques, effective and efficient method of gain tuning for a number of systems and controller types [25][9][10].

Due to their potent optimization property, PSO is currently being investigated for the development of adaptive or self-tuning FOSMC controllers. In this paper, a PSO is select to tune the parameters of the $FOSMC$ controllers of quadcopter altitude and attitude control gain automatically. This fitness function is composed of several parameters extracted from

both regulation and trajectory tracking the non-linear and coupled dynamics of a quadcopter. The position of each particle corresponds to a setting of the fractional order sliding mode controller. The speed of each particles corresponds to this parametrized modification between the two simulations. The velocity and position of each particle/variables are randomly initialized. All particles move in the search space, and the performance of each particle is periodically evaluated.[25][9].

- PSO is an intelligently select optimal parameters from N particles.
- The initialization matrix contains N particles dispersed in a D-dimensional search space.
- PSO requires a fitness function relevant to the particles position.
- Each particle i stores its best position $P_{bi}(t+1)$ and the best solution in its vicinity $g_b(t+1)$, which is the position of the particle that has the smallest fitness value in the swarm.
- The mechanism of displacement of each particle is managed by three rules.
  - First, the particle tends to follow the direction of its current velocity.
  - Second, it wants to move toward its best position.
  - Finally, it tends to move to the best position reached by its neighbours

4.3.1 Algorithm of PSO

1. Initialize a population array of particles with random positions and velocities on D dimensions in the search space.
2. For each particle, evaluate the desired optimization to minimize fitness function in D variables in the swarm.
3. Compare particle's fitness evaluation with its personal pest (p_{besti}) If current value is better than personal pest (p_{besti}) then reset p_{besti} equal to the current value, and p_{besti} equal to the current location in D dimensional space.
4. Compare p_{best} of particles with each other and update the swarm global best location with the greatest fitness and update g_best .

5. update the velocity and position of the particle according to equations below respectively.
   The new velocity matrix v_{ij} and position matrix x_{ij} of the particle $i = 1, 2...N$ in the search space of dimension D, with $j = 1, 2...D$ are

$$v_{ij}(t+1) = wv_{ij}(t) + R_1C_1\left(Pb_{ij}(t) - x_{ij}(t)\right) + R_2C_2\left(gb_{ij}(t) - x_{ij}(t)\right) \tag{4.29}$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1) \tag{4.30}$$

Evaluate the objective /fitness functionf_i and update p_{besti} and g_{best}

$$\begin{cases} p_{besti} & = X_i \\ f_{pbesti} & = f_i \end{cases} \quad \text{if } f_i < f(p_{besti}) \tag{4.31}$$

$$\begin{cases} g_{best} & = p_{besti} \\ f_{gbest} & = f(p_{besti}) \end{cases} \quad \text{if } f(p_{besti} < f(g_{best})) \tag{4.32}$$

Where

- p_{besti} is the best position found by the particle i.
- g_{best} is the best position found by the neighbourhood.
- w, C_1 and C_2 are inertial and acceleration coefficients
- R_1 and R_2 are random variables generated from a uniform distribution in $[0, 1]$

The update velocity of equation 4.29 have three parts:

Part one:- Momentum part $(wv_ij(t))$ and serve as memory of previous flight, prevent the particle from drastically changing direction. Displacement physical component of which w is a variable parameter tolerating to control the displacement of the particle at the next iteration.

Part two:-Cognitive part $R_1C_1(Pb_{ij}(t) - x_{ij}(t))$ oblige to calculate the performance of i^{th} particle relative to its past performance, particles are drawn back to its best position.

Part three:-Social part $R_2C_2(gb_{ij}(t) - x_{ij}(t))$ quantifies the performance of i^{th} particle relative to its neighbours. Particles are drawn towards best position determined by the group. The degree of attracting each particle towards the best position of global best position respectively represented by two acceleration coefficients C_1 and social C_2. If

C_1 equal to C_2 equal to zero particles move in the same direction until it reaches the search space. If C_1 greater than zero and C_2 equal to zero, particles perform local search, whereas C_1 equal to C_2 particles attracts towards the average of $p_{best}i$ and g_{best} .The good exploration of particles in the search space is guaranteed by the coefficients R_1 and R_2, which are two random, numbers drawn uniformly in the interval $[0, 1]$

4.3.2 Fitness function

Quantification of the optimization performance of the quadrotor's responses using the proposed approach is achieved through a performance index [10]. These performance index also known as fitness function (objective) is defined in order to minimize the differences between the desired and the controlled outputs responses of the quadrotor. For these thesis, to minimize the error of the position and orientation of quadcopter trajectory tracking. This fitness function preferred by the control engineering discipline include Integral Square-Error (ISE) ,Integral Time Square-Error (ITSE) ,Integral Absolute-Error (IAE) and Integral Time Absolute Error (ITAE) of equation 4.33

$$\begin{aligned} ISE &= \int_0^\infty e^2(t)dt \\ ITSE &= \int_0^\infty te^2(t)dt \\ IAE &= \int_0^\infty |e(t)|dt \\ ITAE &= \int_0^\infty t|e(t)|dt \end{aligned} \tag{4.33}$$

The characteristic of each fitness function listed in equation 4.33 is different. ISE index is used to penalizes large errors heavily and small errors lightly. A system designed by this criterion tends to show a rapid decrease in a large initial error to get fast and oscillatory response of the system. ITSE emphasis on initial errors and heavily penalizes errors occurring late in the transient response of the system, whereas IAE Penalizes control errors. System's designed using ITAE has small overshoots and minimum damped oscillations; any large initial error is penalized lightly whilst errors occurring later in the response are penalized heavily during ITAE, leads to Penalizes long settling time and control errors. The ITAE performance index used in this study.

Chapter 5

Simulation Results and Discussion

5.1 Introduction

This chapter is presents Simulink model of the quadcopter system modelled mathematically in chapter three with the simulation results of the closed loop control of the system using FOSM controller with MATLAB/Simulink and the gain obtained by using automatic tuning of PSO. Then the model of the UAV is developed in computer Simulink model without disturbance and with disturbance. This model is useful for simulating and observing the behavior of the system with the action of the proposed control system, and the result of the simulation is presented in the form of graphs. A simulation study of using the proposed controller for an agricultural monitoring UAV is performed by minimize the trajectory error and appearance of uncertainties for efficient stabilization and trajectory tracking. Lastly, Simulation results are presented and discussed to indicate the effectiveness of the proposed quadcopter based on position and orientation controlling of the flight system at different trajectories. The proposed control system is represented by Figure 4.1 is designed for simulation by MATLAB/Simulink model. Finally, to verify the validity and efficiency of the proposed controller, Appling different trajectories with disturbance like helical, infinity trajectory and rectangular trajectories.

The Particle Swarm optimization of the FOSMC gain was performed with various combinations of total iterations T, numbers of particles N, with the cost functions. Explained in chapter four.

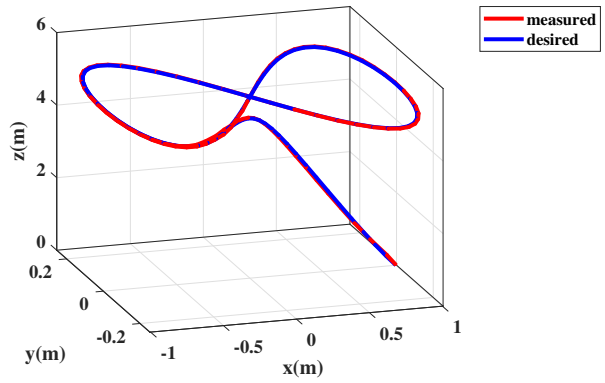
5.2 Infinity trajectory tracking

An Infinity trajectory was used as a tracking reference in the space as follows.The input trajectories along x and y is sinusoidal shape which is $x_r = cos(t)$, $y_r = \frac{sin(2t)}{4}$ and z-axis is linear $z_r = (15/4) * t^2 - (5/4) * t^3$ for $t < 2$ otherwise constant $z_r = 5$ and $\psi_r = 0.8$ rad. Sample controller gains obtained by PSO are shown in Tables 5.1 for $N = 10, F = 10$

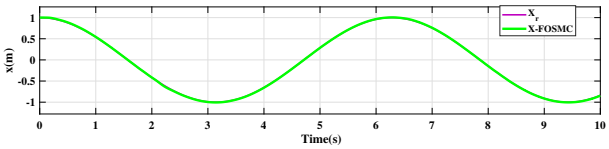
Table (5.1) An Infinity trajectory tracking controller gains by pso

gain	**value**	**gain**	**value**
c_x	25	c_ϕ	30
β_x	0.25	β_ϕ	0.25
k_x	1.6	k_ϕ	1.6
c_y	25	c_θ	30
β_y	0.25	β_θ	0.25
k_y	1.6	k_θ	1.6
c_z	25	c_ψ	30
β_z	0.25	β_ψ	0.25
k_z	1.6	k_ψ	1.6

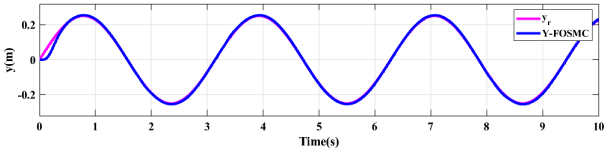
The resulting $3D$ trajectory tracking response is presented in figure 5.1a where the tracking positions along x, y and z-axis are presented in figure 5.1. and the the generated reference signal from outer loop with their trajectory output and yaw angle trajectory is shown in figure 5.2. From the obtained responses it is shown that the proposed FOSMC tuned PSO scheme is able to guarantee accurate reference tracking.



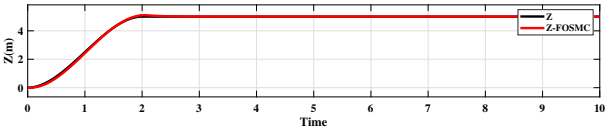
(a) 3d Plot of the Infinity trajectory being tracked



(b) Infinity trajectory position along x-axis.



(c) Infinity trajectory position along y-axis.



(d) Infinity trajectory position along z-axis.

Figure (5.1) Infinity trajectory tracking response and positions along x,y,z-axis.

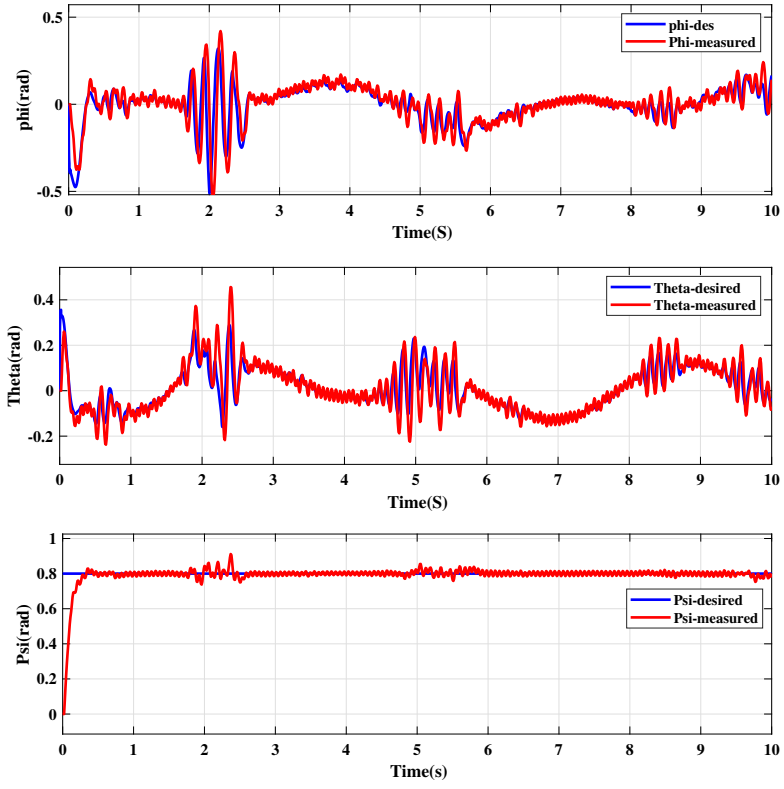


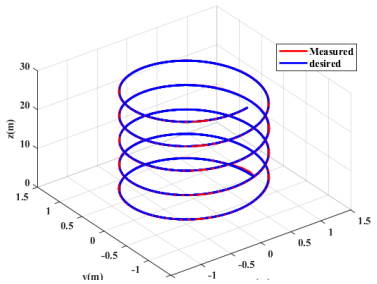
Figure (5.2) Infinity trajectory tracking of attitude .

5.3 Helical trajectory of quadcopter

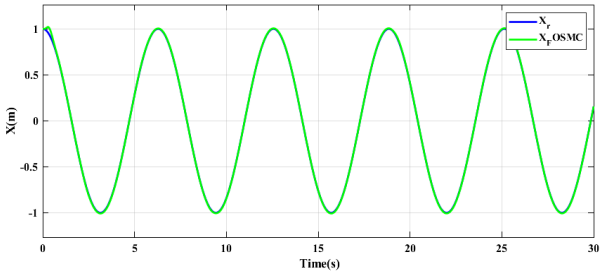
The resulting 3D helical trajectory tracking response and positions are shown fin figure5.3.Applying PSO technique to obtain optimal gains of FOSM controller for helical trajectory tracking by running many times and select the bast performance. The best results given in table5.2

Table (5.2) Helical trajectory controller gains by pso

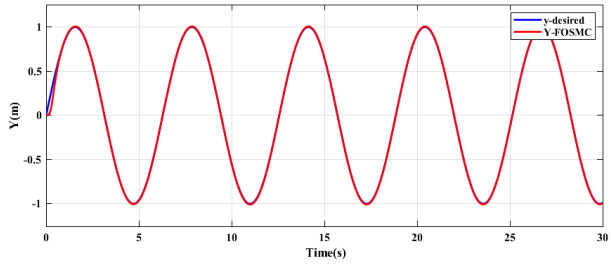
gain	value	gain	value
c_x	15	c_ϕ	10
β_x	0.3	β_ϕ	0.2
k_x	1.6	k_ϕ	1
c_y	15	c_θ	10
β_y	0.3	β_θ	0.2
k_y	1.6	k_θ	1
c_z	15	c_ψ	10
β_z	0.3	β_ψ	0.2
k_z	1.6	k_ψ	1



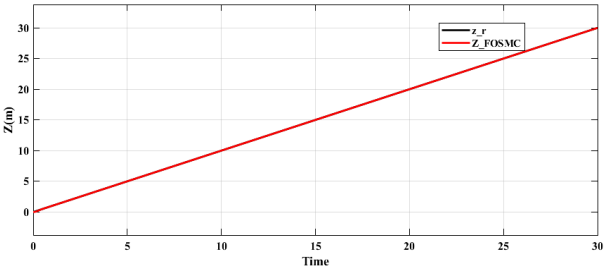
(a) Helical trajectory tracking



(b) Position along x-axis.



(c) Position along y-axis.



(d) Position along z-axis.

Figure (5.3) Helical trajectory tracking response and positions along x,y,z-axis.

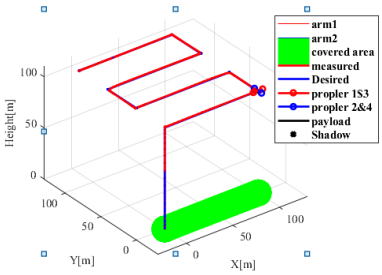
5.4 Square wave path trajectory

Unmanned Aerial Vehicles' (UAVs) capabilities offer the potential to capture the image platforms for real-time agricultural monitoring, use these trajectories to cover the entire area.the input trajectories written in appendix D.3.This experimental simulation on the quadrotor to test the efficiency of FOSMC for performing those tasks. The vehicle tracks a square wave trajectory with automatic gain tuning by PSO.
Sample controller gains obtained by PSO are shown in Tables 5.3 for $N = 10, F = 10$

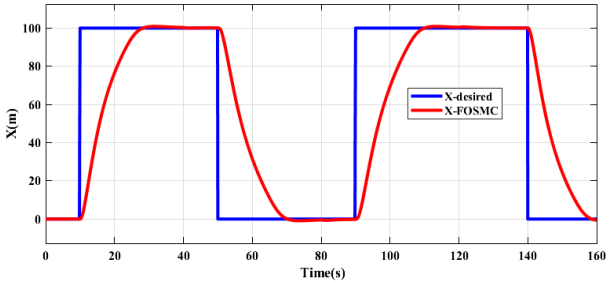
Table (5.3) A Square wave trajectory controller gains by pso

gain	**value**	**gain**	**value**
c_x	0.36	c_ϕ	10
β_x	0.15	β_ϕ	0.2
k_x	1.8	k_ϕ	1
c_y	0.36	c_θ	10
β_y	0.15	β_θ	0.2
k_y	1.8	k_θ	1
c_z	0.36	c_ψ	10
β_z	0.15	β_ψ	0.2
k_z	1.8	k_ψ	1

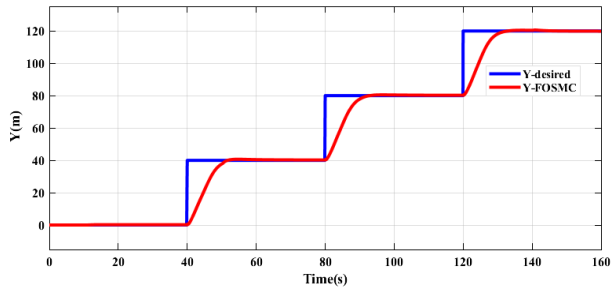
The response of the system in $3D$ and tracking response of each positions for the respective inputs are shown in figure 5.4 . But the control signal in each curve is maximum ,to overcome such difficulty let us use differentiable polynomial inputs for x-axis as shown in figure 5.5.



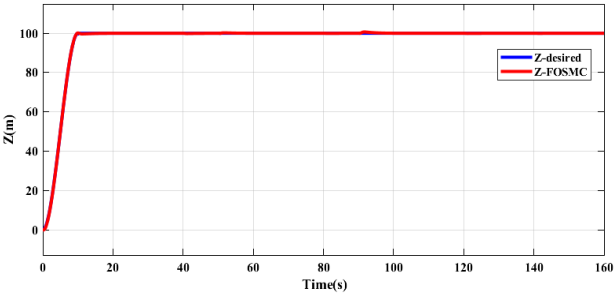
(a) Square wave path trajectory



(b) Position along x-axis.



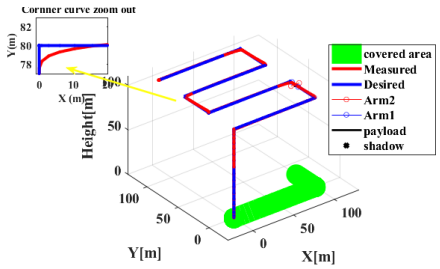
(c) Position along y-axis.



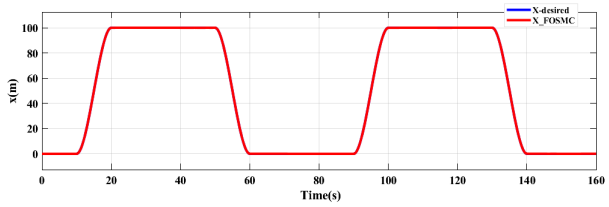
(d) Position along z-axis.

Figure (5.4) Square wave response and positions trajectory for step input.

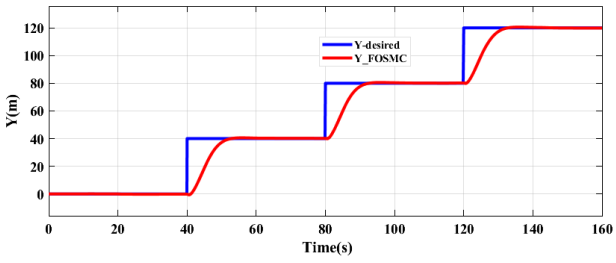
Figure 5.5 shows the response of the system in $3D$ and tracking response of each positions (x, y, z) obtained by the proposed method. The input trajectories of third order polynomial function ,stare case and step input along x,y and z-axis respectively.



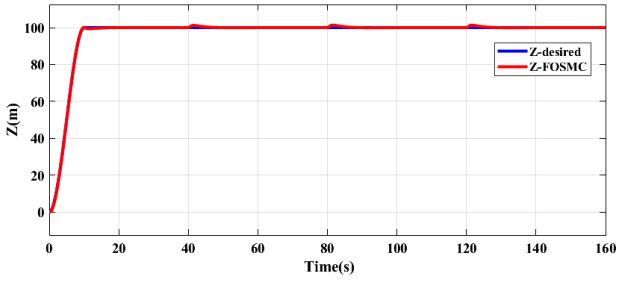
(a) Square wave path trajectory



(b) Position along x-axis.



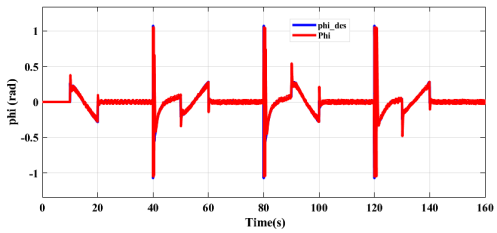
(c) Position along y-axis.



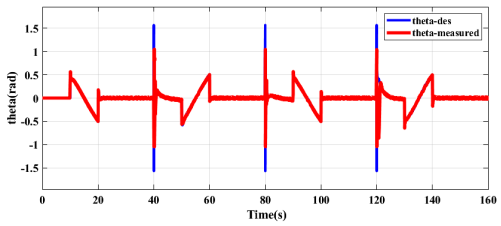
(d) Position along z-axis.

Figure (5.5) Square wave response and positions trajectory for polynomial along x axis input

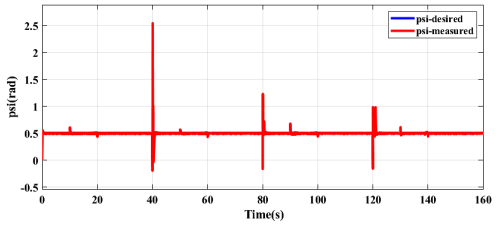
It can be noticeably observed From Figure **??** and Table 5.3 that the FOSMC autotuned controllers using PSO have good performances (reduced errors) for positions (x, y, z) control. Responses for desired and controlled roll (ϕ) and pitch (θ) attitudes are illustrated in Figure 5.6



(a) Roll angle trajectory



(b) Pitch angle trajectory.

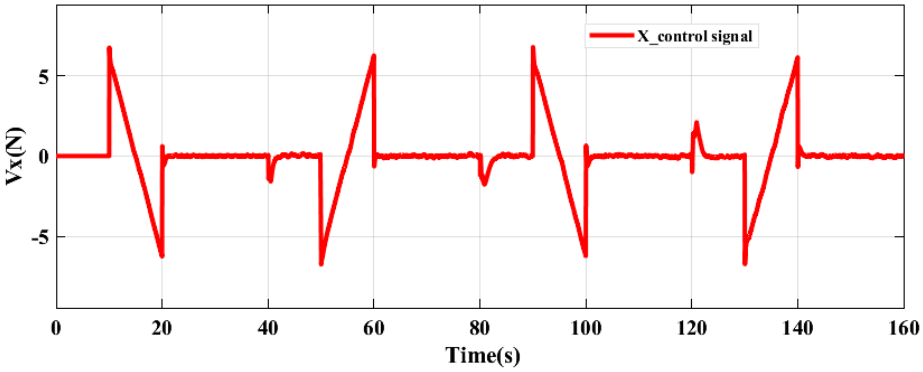


(c) Yaw angle trajectory.

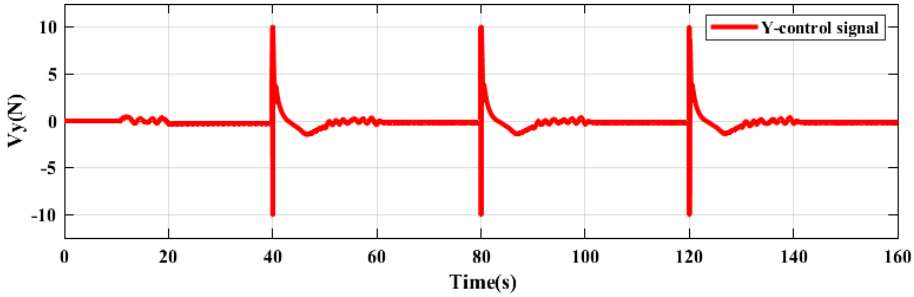
Figure (5.6) Euler angle response for Square wave trajectory .

From figure 5.6 we can observe that the measured angles follow the desired angles generated by the outer control loop of the control system and the yaw angle tracks the reference.

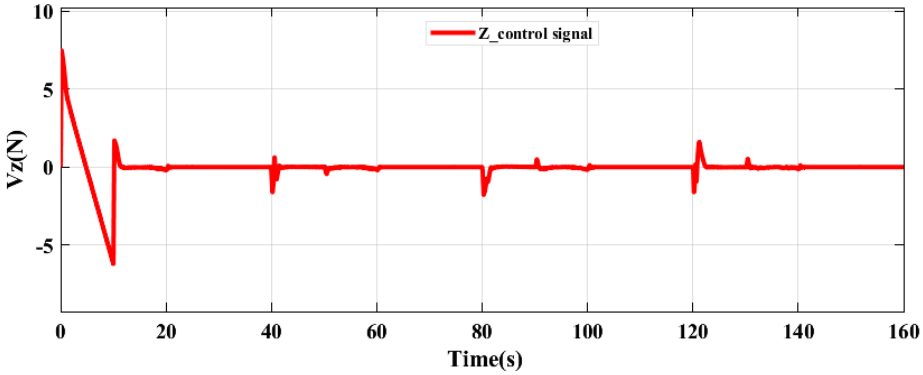
The outer loop or virtual controller signals Vx, Vy, and Vz which are used to determine the desired roll ϕ_r and pitch θ_r and attitude controller $U1$ are shown in figure 5.7 . The controller signal amplitude is vary based on the input trajectory. At the curve of the given trajectory and during applied disturbance the quadcopter needs maximum control signal to follow the path as shown in figure 5.7.



(a) Virtual controller along x-axis



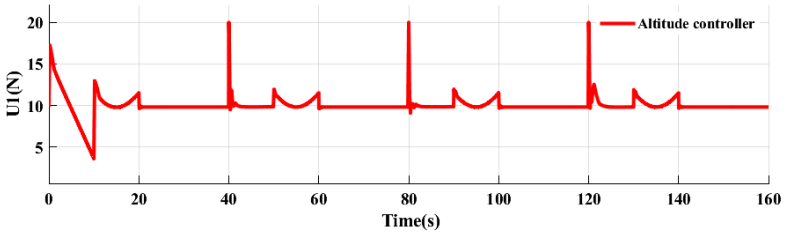
(b) Virtual controller along y-axis



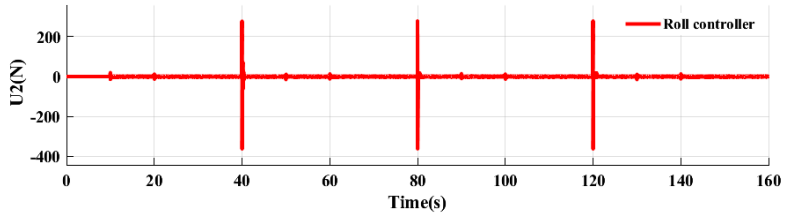
(c) Virtual controller along z-axis

Figure (5.7) Virtual controller response for position trajectory.

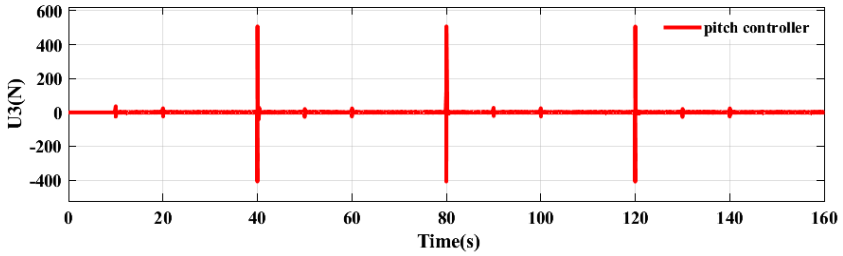
Inner loop controller and altitude controllers given by figure 5.8. Altitude controller U_1 of figure 5.8a and yaw angle controller U_4 of figure 5.8d controls the desired altitude and yaw angle given by the task manager respectively, Where as the attitude roll angle controller U_2 of figure 5.8b and pitch angle controller U_3 of figure 5.8c controls the desired trajectory generated from the position by outer controller signals respectively.



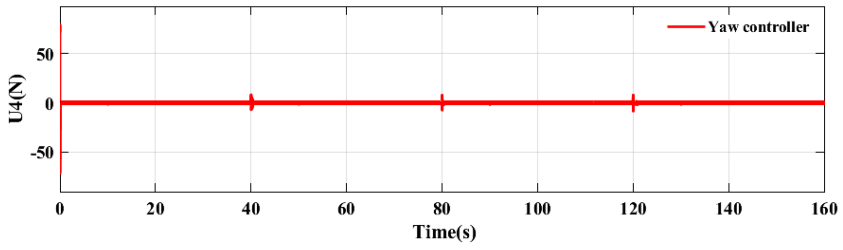
(a) Quadcopter altitude controller effort



(b) Quadcopter Roll angle controller effort.



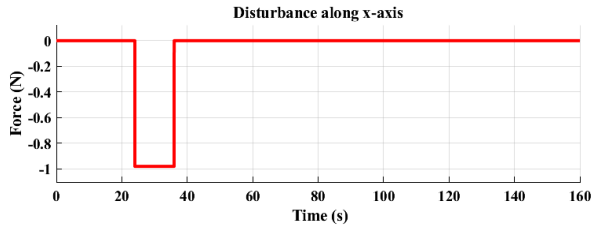
(c) Quadcopter pitch angle controller effort



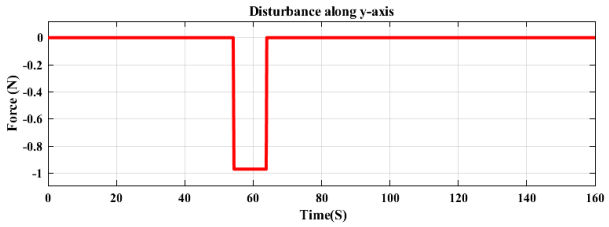
(d) Quadcopter yaw angle controller effort.

Figure (5.8) Altitude and Attitude controllers effort of quad-copter

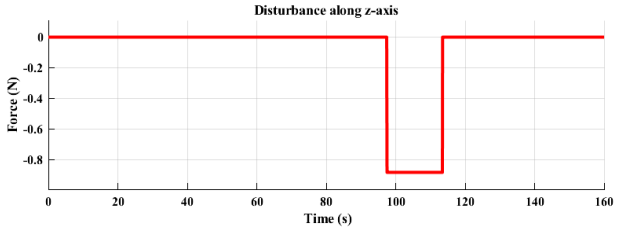
In order to show the efficiency and robustness of FOSMC scheme in tracking paths apply unknown disturbance utilized in real life applications tracking response for a square wave reference path, to monitoring and exploration of agriculture. The system response of the proposed controllers under disturbances in $3D$ is presented in the Figure 5.9d. From the simulation result we observe that the suggested controller achieve better performance of the desired trajectory tracking in the presence of disturbances. The external disturbance modeled in this simulation is a random signal with a force of different amplitude along x,y, and z positions at time intervals shown in figure 5.9a, 5.11c and 5.11d.



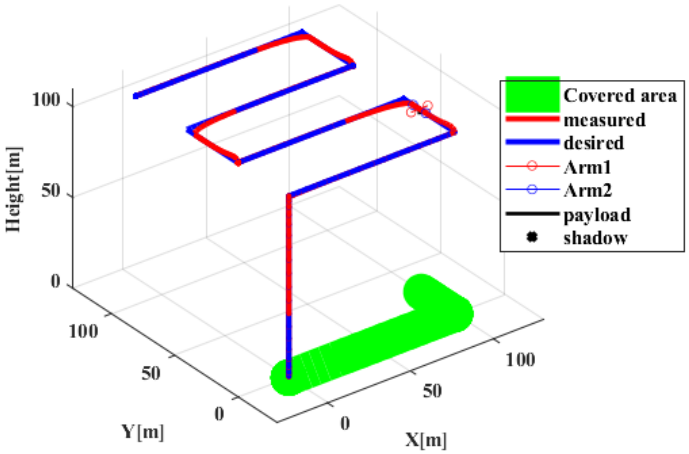
(a) Applied disturbance along x-axis.



(b) Applied disturbance along y-axis.



(c) Applied disturbance along z-axis.

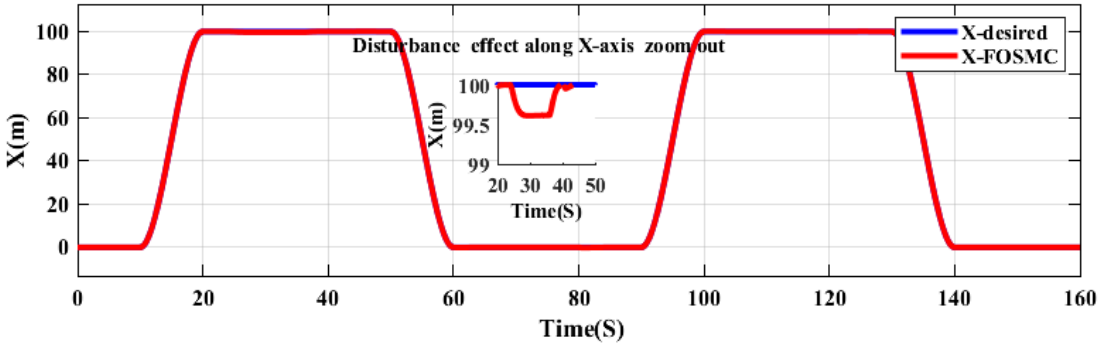


(d) Square wave response with disturbance

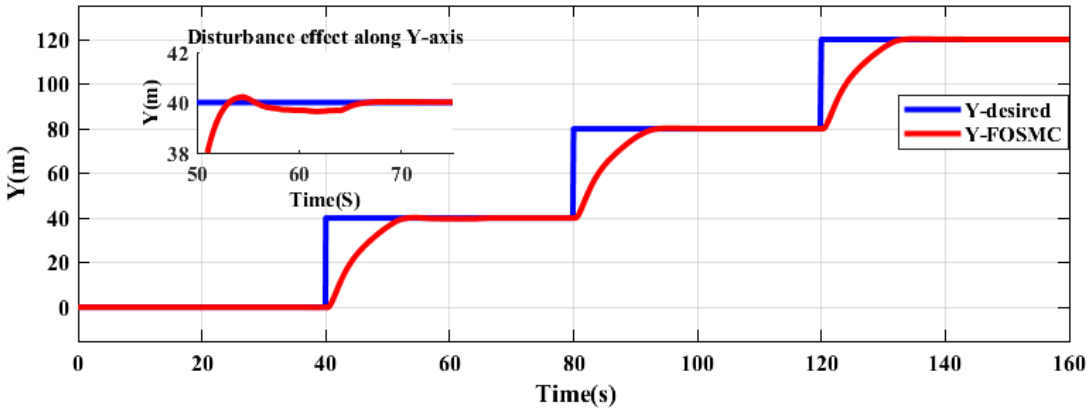
Figure (5.9) Square wave response and applied disturbance

The effect of disturbance and range of ability of the controller for disturbance rejection

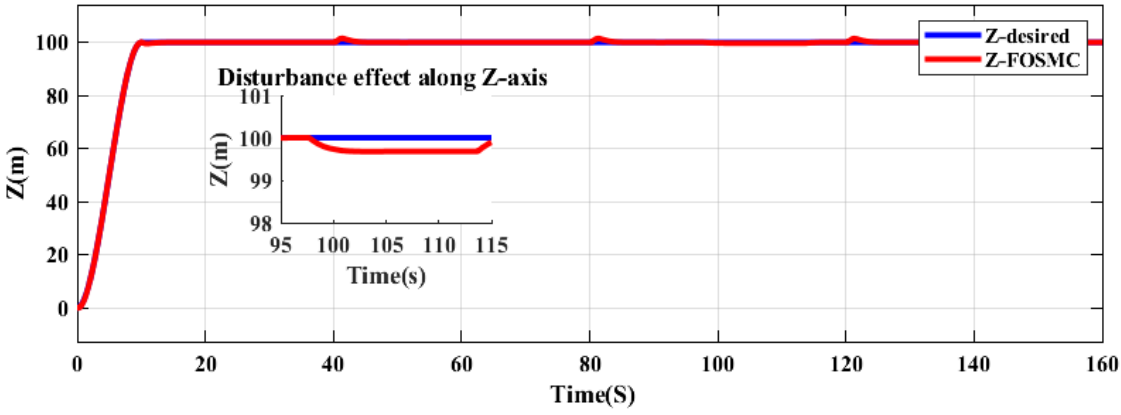
along quadcopter position is shown in figure 5.10. It shows that the FOSMC ensures to stabilize the system and the position convergence to the desired values. Even if we apply higher magnitude of force as disturbance on the dynamics of the quadcopter, FOSMC reject the disturbance and the errors converge to zero.



(a) Effect of disturbance along x-axis.



(b) Effect of disturbance along y-axis.

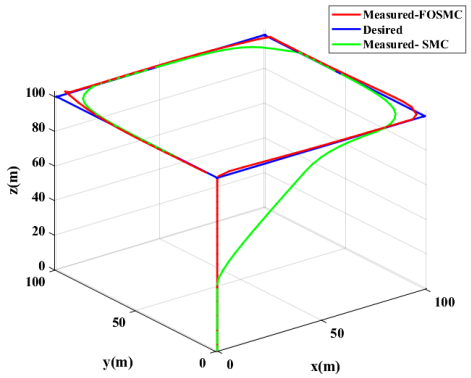


(c) Effect of disturbance along z-axis.

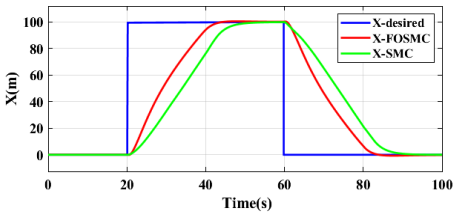
Figure (5.10) Effect of disturbance for positions of UAV

5.5 Comparison of classical SMC with FOSMC

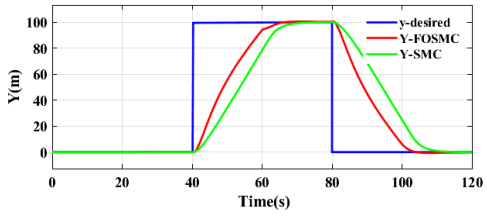
To track a square wave of altitude 100 meters above the ground with respective positions along the x,y z-axis, the FOSMC method could manipulate all state variables to the desired reference signals with a faster convergence rate compared to the conventional SMC method. The simulation result for the Fractional-order sliding mode controller shows good performance whereas the conventional sliding mode controller gives a slow response Introduce steady-state error in the system and lead to more deviations from the given trajectory.



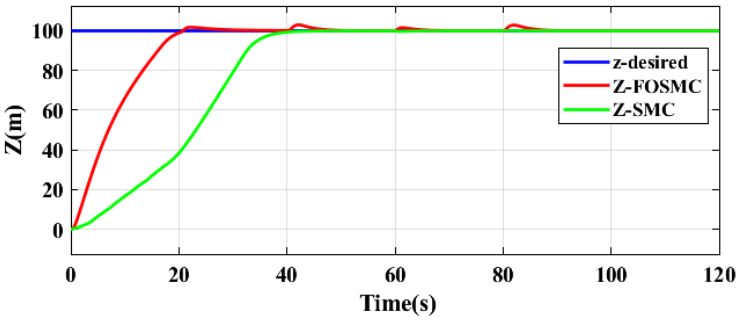
(a) Comparison of SMC with FOSMC in rectangular trajectory.



(b) Position along x-axis.



(c) Position along y-axis.



(d) Position z-axis

Figure (5.11) Comparison of SMC with FOSMC in rectangular trajectory.

Chapter 6

Conclusion and Future Works

6.1 Conclusion

In this paper, a hybrid modeling approach of the quadrotor dynamics is established through Newton–Euler methods and the design of fractional-order Sliding mode control using MATLAB software to overcome the performance of the trajectory tracking. Design of a FOSMC for complete six DOF outputs (positions (x, y, z)) and attitudes (ϕ, θ, ψ). Particle Swarm Optimization (PSO) are applied for optimal tuning of the controllers' parameters of the FOSMC controllers of quadcopter altitude and attitude control gain automatically based on fitness function which is composed of several parameters extracted from trajectory tracking error of quadcopter.

FOSMC method could manipulate all state variables to the desired reference signals with a faster convergence rate compared to the conventional SMC method. The simulation result for the Fractional-order sliding mode controller shows good trajectory tracking whereas the conventional sliding mode controller gives a slow response introduces steady-state error in the system and leads to more deviations from the given trajectory. FOSMC leads better performance to eliminate the tracking errors and be able to keep the quadrotor on the desired way than convectional SMC.

The simulation results show that the suggested PSO tuned FOSMC can achieve robust performance subject to the complex disturbances and leads tracking errors of all the system state variables tends to zero. Even if we apply higher magnitude of force as disturbance on the dynamics of the quadcopter, FOSMC resist the disturbance effects and provide responses close to the desired ones. This robustness and effectiveness are justified by applying different trajectories like rectangular, infinity, square wave, and helical trajectories. The ability of quadcopter capabilities to monitor and explore agriculture by following the determined trajectories observed in the 3D plan.

6.2 Future work

The models built and studied in this thesis focuses only on the performance of the agricultural monitoring quadcopter, robustness and effectiveness of the controller based on minimizing errors to track the desired trajectories and automatic controller gain tuning methods by applying different paths The author recommend interested researchers to design and analyse

1. Obstacle avoidance and Optimum energy for flight of an agricultural quadcopter.
2. Hardware implementation of the modelled quadcopter.

References

[1] G. Sylvester, *E-agriculture in action: Drones for agriculture*. Food and Agriculture Organization ofn the United Nations and International . . . , 2018.

[2] T. Bresciani, "Modelling, identification and control of a quadrotor helicopter," *MSc theses*, 2008.

[3] L. Derafa, A. Ouldali, T. Madani, and A. Benallegue, "Four rotors helicopter yaw and altitude stabilization." in *World Congress on Engineering*. Citeseer, 2007, pp. 148–152.

[4] A. Reizenstein, "Position and trajectory control of a quadcopter using pid and lq controllers," 2017.

[5] R. Tesfaye, "Modeling and control of a quad-rotor unmanned aerial vehicle at hovering position," Ph.D. dissertation, Addis Ababa University, 2012.

[6] M. Usman, "Quadcopter modelling and control with matlab/simulink implementation," 2020.

[7] E. Kuantama, 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, pp. 519–532, 2017.

[8] S. Madruga, A. Tavares, G. Basso, T. Nascimento, and A. Brito, "A pso-based tuning algorithm for quadcopter controllers," 01 2018.

[9] N. El Gmili, M. Mjahed, A. El Kari, and H. Ayad, "Particle swarm optimization based proportional-derivative parameters for unmanned tilt-rotor flight control and trajectory tracking," *Automatika*, vol. 61, no. 2, pp. 189–206, 2020.

[10] ——, "Particle swarm optimization and cuckoo search-based approaches for quadrotor control and trajectory tracking," *Applied Sciences*, vol. 9, no. 8, p. 1719, 2019.

[11] T. T. Mac, C. Copot, T. T. Duc, and R. De Keyser, "Ar. drone uav control parameters tuning based on particle swarm optimization algorithm," in *2016 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)*. IEEE, 2016, pp. 1–6.

[12] İ. C. DİKMEN, T. KARADAĞ, and C. YEROĞLU, "Multi-parameter optimization of sliding-mode controller for quadcopter application," *Computer Science*, vol. 3, no. 1, pp. 14–28, 2018.

[13] M. A. Rendón and F. F. Martins, "Path following control tuning for an autonomous unmanned quadrotor using particle swarm optimization," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 325–330, 2017.

[14] K. Runcharoon and V. Srichatrapimuk, "Sliding mode control of quadrotor," in *2013 The International Conference on Technological Advances in Electrical, Electronics and Computer Engineering (TAEECE)*. IEEE, 2013, pp. 552–557.

[15] S. Sridhar, R. Kumar, M. Radmanesh, and M. Kumar, "Non-linear sliding mode control of a tilting-rotor quadcopter," in *Dynamic Systems and Control Conference*, vol. 58271. American Society of Mechanical Engineers, 2017, p. V001T09A007.

[16] V. T. Hoang and Q. H. Pham, "Adaptive super-twisting second-order sliding mode for attitude control of quadcopter uavs," *arXiv preprint arXiv:1803.07690*, 2018.

[17] K. Can, K. Orman, A. BAŞÇİ, and A. Derdiyok, "A fractional-order sliding mode controller design for trajectory tracking control of an unmanned aerial vehicle," vol. 26, pp. 4–10, 08 2020.

[18] O. V. Dhakad and V. Kumar, "Fractional order sliding-mode controller for quadcopter," in *Advances in Interdisciplinary Engineering*. Springer, 2019, pp. 381–392.

[19] T. Huang, D. Huang, Z. Wang, and A. Shah, "Robust tracking control of a quadrotor uav based on adaptive sliding mode controller," *Complexity*, vol. 2019, 2019.

[20] R. Niemiec and F. Gandhi, "A comparison between quadrotor flight configurations," 2016.

[21] S. M. ME, R. Maguteeswaran, N. G. BE, and G. Srinivasan, "Quadcopter uav based fertilizer and pesticide spraying system," *International Academic Research Journal of Engineering Sciences, ISSN*, no. 2414-6242, 2016.

[22] I. Petráš, *Fractional-order nonlinear systems: modeling, analysis and simulation*. Springer Science & Business Media, 2011.

[23] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-international conference on neural networks*, vol. 4. IEEE, 1995, pp. 1942–1948.

[24] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization," *Swarm intelligence*, vol. 1, no. 1, pp. 33–57, 2007.

[25] D. V. Quoc, N. D. T. Minh, T. T. Thi, L. N. Tien, M. N. Quang, and B. N. T. Thanh, "Tuning pid controller bases on random search algorithms."

[26] A. Nemati and M. Kumar, "Modeling and control of a single axis tilting quadcopter," in *2014 American Control Conference*. IEEE, 2014, pp. 3077–3082.

Appendices

Appendix A

Rotation matrix coriols force and desired angles derivations

A.1 Derivation of rotational matrix

The rotation matrix that transforms a linear quantity from earth fixed coordinates to body fixed coordinate.The rotation matrix is the product of rotation around the z-axis followed by rotation around y-axis and finally followed by rotation around x-axis. All three matrices multiplication of the three standard rotations matrices, the selected order of rotations of $R_Z - R_Y - R_X$ Sequentially.

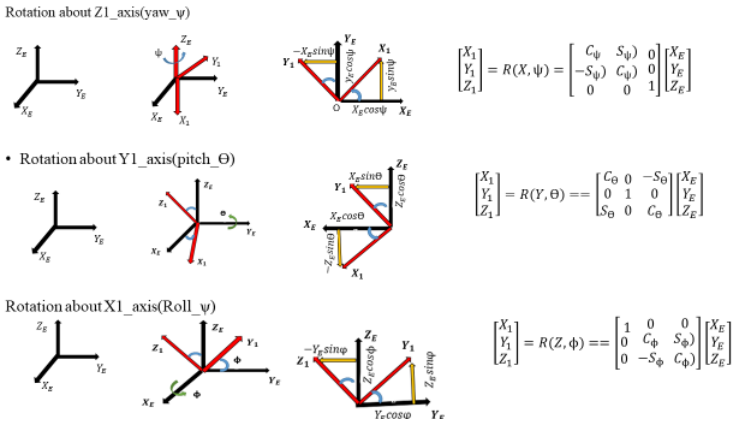


Figure (A.1) Inertial frame to body frame transformation matrix

The transformation matrix between two coordinate systems is orthogonal. $R(x,\phi)$, $R(y,\theta)$ and $R(z,\psi)$ denote rotation matrices produced by the inertial coordinate frame rotating roll angle ϕ, pitch angle θ, and yaw angle ψ around x, y, and z-axes, respectively as

follows.

yaw angle ψ rotates with Z axis.

$$R_{z,\psi} = \begin{bmatrix} cos\psi & sin\psi & 0 \\ -sin\psi & cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

pitch angle θ rotates with y axis.

$$R_{y,\theta} = \begin{bmatrix} cos\theta & 0 & -sin\theta \\ 0 & 1 & 0 \\ sin\theta & 0 & cos\theta \end{bmatrix}$$

Roll angle ϕ rotates with x axis.

$$R_{x,\phi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\psi & -sin\psi \\ 0 & sin\psi & cos\psi \end{bmatrix}$$

Orderly combination of these three matrices listed on the above figure gives the inertial to body frame transformation matrix. Order of the operations is important.

$$\overset{B}{_E}R = \begin{bmatrix} c\theta c\psi & c\theta s\phi & -s\theta \\ s\phi s\theta c\psi - c\phi s\psi & s\phi s\theta s\psi + c\phi c\theta & c\theta s\phi \\ c\phi s\theta c\psi + s\phi s\psi & s\phi s\theta c\phi - c\psi s\phi & c\phi c\theta \end{bmatrix} \tag{A.1}$$

A.2 Derivation of coriols force

Let the inertial set of unit vectors along the direction,$b = \begin{bmatrix} b_1 & b_2 & b_3 \end{bmatrix}$, which has an instantaneous angular velocity ,$\omega = \begin{bmatrix} p & q & r \end{bmatrix}$, we want to calculate derivative of linear velocity vector v with respect to body frame [26].

$$\frac{dv}{dt}_{inertial} = \frac{d(ub_1 + vb_2 + wb_3)}{dt}$$

Using chain rule:

$$\frac{dv}{dt}_{inertial} = b_1\frac{du}{dt} + u\frac{db1}{dt} + b_2\frac{dv}{dt} + v\frac{d_{b2}}{dt} + b_3\frac{dw}{dt} + w\frac{d_{b3}}{dt}$$

.

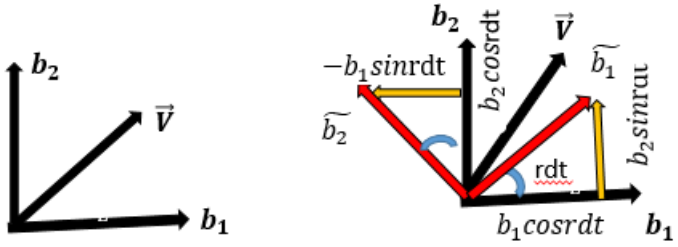
Now,the question is that how to obtain $\frac{d()}{dt}\begin{bmatrix} b_1 & b_2 & b_3 \end{bmatrix}$

The derivative of the unit vector for each rotation W rotation of one axis at a time we can represent the change from any initial set of unit vectors $\begin{bmatrix} b_1 & b_2 & b_3 \end{bmatrix}$ to a new set of unit

vectors $\begin{bmatrix}\hat{b_1} & \hat{b_2} & \hat{b_3}\end{bmatrix}$ 'across some change in time dt with an angle Θ. The relation between angular velocity with those angle is expressed as:

$$w = d\theta/dt$$
$$\theta = w_i dt \text{ ,where } i = \begin{bmatrix}b_1 & b_2 & b_3\end{bmatrix}$$

Let us rotate first along z-axis unit vector:



(a) unit vector axis (b) z-axis unit vector rotation

Figure (A.2) Unit vector rotation along z-axis

$$\begin{bmatrix}\hat{b}_1\\ \hat{b}_2\\ \hat{b}_3\end{bmatrix} = \begin{bmatrix}C(rdt) & S(rdt) & 0\\ -S(rdt) & C(rdt) & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}b_1\\ b_2\\ b_3\end{bmatrix}$$

Rewrite the above equation to return the rotation back from $\begin{bmatrix}\hat{b}_1 & \hat{b}_2 & \hat{b}_3\end{bmatrix}$ *to* $\begin{bmatrix}b_1 & b_2 & b_3\end{bmatrix}$ we can express bye transpose of the square matrix as:

$$\begin{bmatrix}b_1\\ b_2\\ b_3\end{bmatrix} = \begin{bmatrix}C(rdt) & -S(rdt) & 0\\ S(rdt) & C(rdt) & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\hat{b}_1\\ \hat{b}_2\\ \hat{b}_3\end{bmatrix}$$

By taking same procedure for about $b_2(y)$ axis and $b_1(x)$ axis we got

$$\begin{bmatrix}b_1\\ b_2\\ b_3\end{bmatrix} = \begin{bmatrix}C(qdt) & 0 & S(qdt)\\ 0 & 1 & 0\\ -S(qdt) & 0 & C(qdt)\end{bmatrix}\begin{bmatrix}\hat{b}_1\\ \hat{b}_2\\ \hat{b}_3\end{bmatrix}$$

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \\ 0 & C(pdt) & -S(pdt) & \\ 0 & S(pdt) & C(pdt) & 0 \end{bmatrix} \begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \\ \hat{b}_3 \end{bmatrix}$$

To simplify using small angle approximation theorem:-we care about the instantaneous change of unit vectors as dt approaches to zero which yields $cos(\theta) \cong 1$ and $sin(\theta) \cong \theta$ Where $c = cos(.)$ and $s = sin(.)$ The rotations along z,x and x-axis written as:

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 1 & -rdt & 0 \\ rdt & 1) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \\ \hat{b}_3 \end{bmatrix}$$

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & qdt \\ 0 & 1 & 0 \\ -qdt & 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \\ \hat{b}_3 \end{bmatrix}$$

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \\ 0 & 1 & -pdt & \\ 0 & pdt & 1 & 0 \\ & & & \end{bmatrix} \begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \\ \hat{b}_3 \end{bmatrix}$$

To find the derivative from the delta over dt calculate delta in unit vector by subtracting the change in unit vector over time dt at time $t0$. $\begin{bmatrix} b_1 & b_2 & b_3 \end{bmatrix}$ and $\begin{bmatrix} \hat{b_1} & \hat{b_2} & \hat{b_3} \end{bmatrix}$ From rotation about Z-direction which is in r-angular velocity

Divide both sides of the above equation by dt and express in matrix form as:

$$\begin{bmatrix} b_1/dt \\ b_2/dt \\ b_3/dt \end{bmatrix} = \begin{bmatrix} 1 & -r & 0 \\ r & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \\ \hat{b}_3 \end{bmatrix}$$

Applying the same procedure, we can get angular velocities along y-axis and x-axis.

$$\begin{bmatrix} b_1/dt \\ b_2/dt \\ b_3/dt \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -p \\ 0 & p & 0 \end{bmatrix} \begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \\ \hat{b}_3 \end{bmatrix}$$

$$\begin{bmatrix} b_1/dt \\ b_2/dt \\ b_3/dt \end{bmatrix} = \begin{bmatrix} 1 & 0 & q \\ 0 & 0 & 0 \\ -q & 0 & 0 \end{bmatrix} \begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \\ \hat{b}_3 \end{bmatrix}$$

Combine all rotations for all axis together in one matrix

$$\begin{bmatrix} b_1/dt \\ b_2/dt \\ b_3/dt \end{bmatrix} = \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix} \begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \\ \hat{b}_3 \end{bmatrix}$$

Which is equal to $wX \begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \\ \hat{b}_3 \end{bmatrix}$ this is coriols equation.

Then back to the chain rule and substitute the values

$$\frac{dv}{dt}_{inertial} = \frac{du}{dt}b_1 + \frac{db_1}{dt}u + \frac{dv}{dt}b_2 + d\frac{b2}{dt}v + d\frac{w}{dt}b_3 + d\frac{b3}{dt}w$$

Substitute on to newton's second law

$$F = ma = mdv/dtF_x = m(\dot{u} + vr + wq)F_y = m(\dot{v} + wp - ur)F_z = m(\dot{w} + uq - vp) \quad \text{(A.2)}$$

Rewrite in matrix form as follows.

$$F = \begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix} = \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & m \end{bmatrix} \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} + \begin{bmatrix} 0 & mw & -mv \\ -mw & 0 & mu \\ mv & -mu & 0 \end{bmatrix} \begin{bmatrix} p \\ q \\ q \end{bmatrix}$$

A.3 Desired angles derivation

To drive the desired angles and altitude controller U_1 from equation 4.21 as follows:

$$\ddot{x} = V_x = cos\phi sin\theta cos\psi + sin\phi sin\psi)\frac{u_1}{m}$$
$$\ddot{y} = V_y = cos\phi sin\theta sin\psi - sin\phi cos\psi\frac{u_1}{m}$$
$$\ddot{z} = V_z = cos\phi cos\theta\frac{u_1}{m} - g$$

rearrange these equation as

$$\frac{U_1}{m} = \frac{V_z + g}{c_\phi c_\theta}$$
$$U_x = \frac{V_z + g}{c_\phi c_\theta}(c_\phi s_\theta s_\psi + s\psi s_\phi)$$
$$c_\phi s_\theta = (\frac{U_x c_\phi c_\theta}{V_z + g})\frac{1}{c_\psi} - s_\psi s_\theta$$

from virtual controller along y axis extract $c_\phi s_\theta$ as

$$c_\phi s_\theta = \frac{U_y c_\phi c_\theta}{V_z + g}\frac{1}{c_\psi} + s_\psi c_\theta$$

equate these equations and multiplied by $c_\phi c_s\psi$

$$s_\phi c_\psi^2 + s_\phi s_\psi^2 = \frac{U_x c_\phi c_\theta s_\psi - U_y c_\phi c_\theta c_\psi}{V_z + g}$$
$$tan\phi = c_\theta\frac{U_x s_\psi - U_y c_\psi}{V_z + g}$$
$$\phi_d = tan^{-1}c\theta_d\frac{U_x s_\psi - U_y c_\psi}{V_z + g}$$

applying the same method and re arranging the above eqution to get θ_d as

$$tan\theta = \frac{U_x c_\psi + U_y s_\psi}{V_z + g}$$
$$\theta_d = tan^{-1}\frac{U_x c_\psi + U_y s_\psi}{V_z + g}$$

Appendix B

Simulink® Complete Model

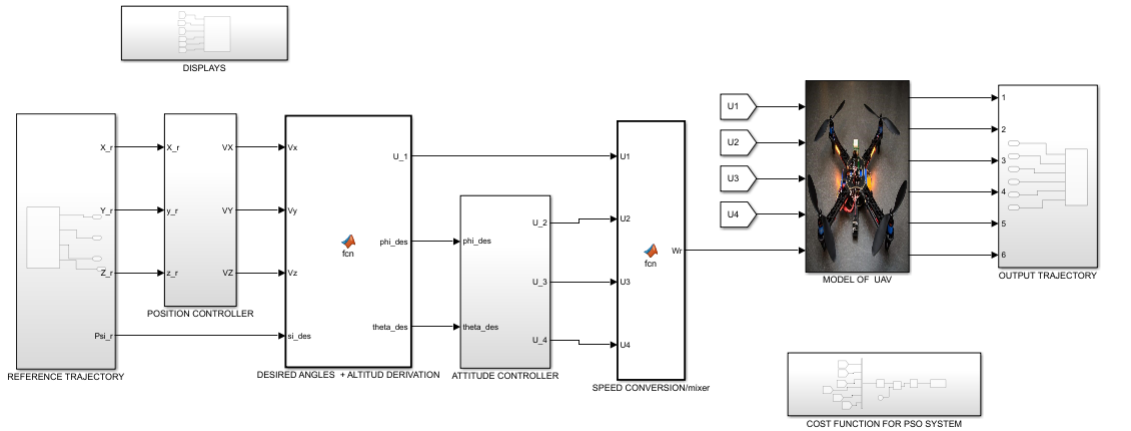
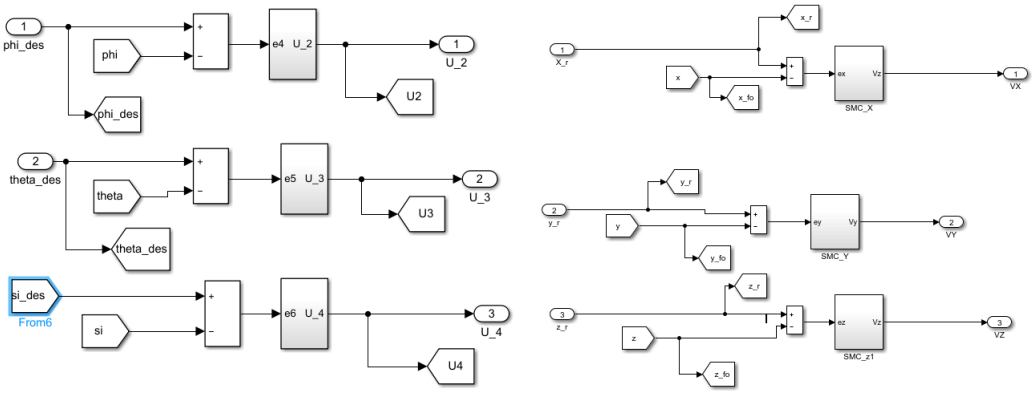


Figure (B.1) SIMULINK® Model of the Complete FOSM controller based quadcopter.

B.1 Attitude and position controllern



(a) Position controller sub block. (b) Attitude controller sub block.

Figure (B.2) Extracting The position and attitude Block of Figure B.1

B.2 Virtual /position controllers along x y and z-axis

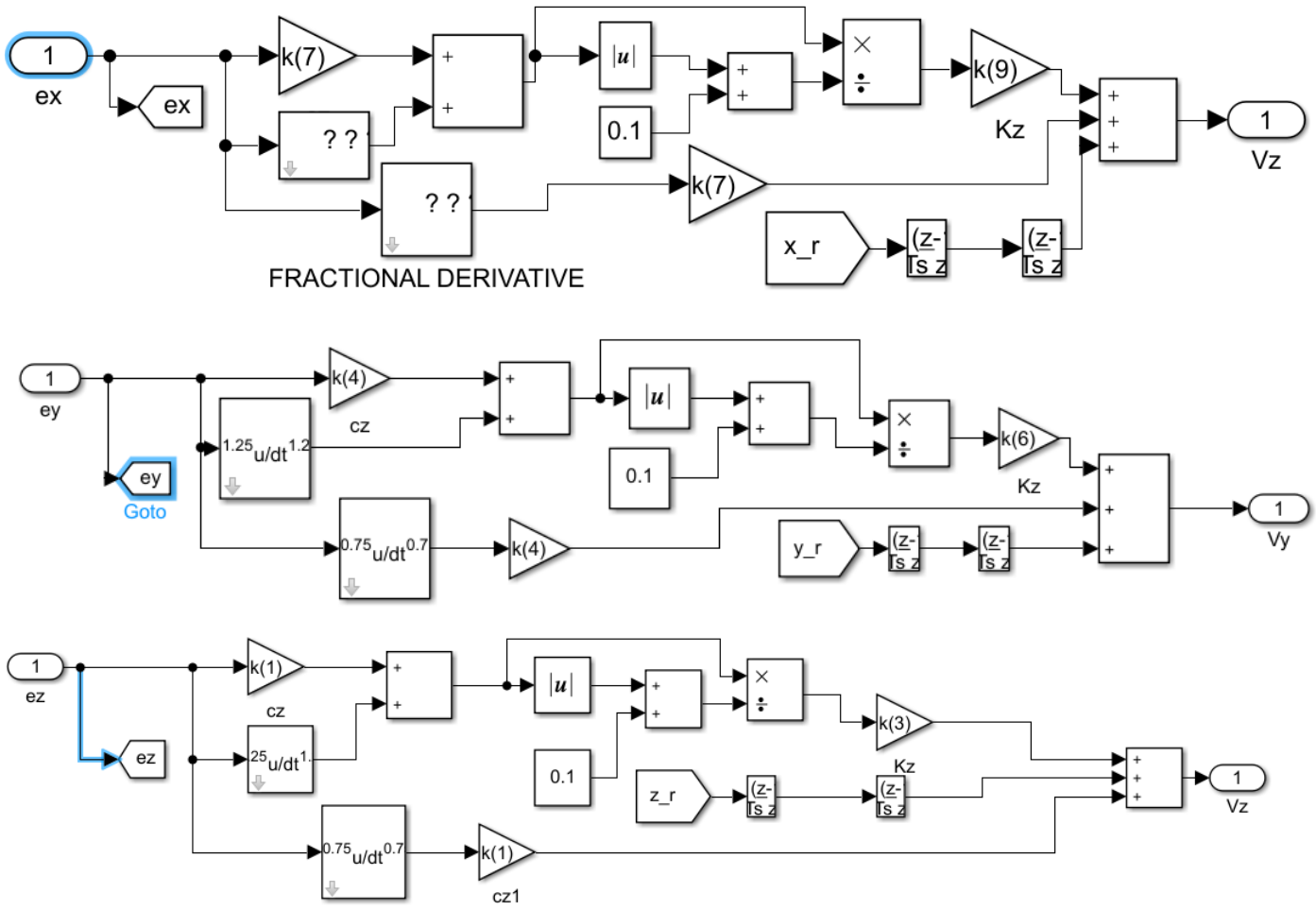


Figure (B.3) Virtual/position controllers extracted from B.2

B.3 Inner Loop Roll movement controller

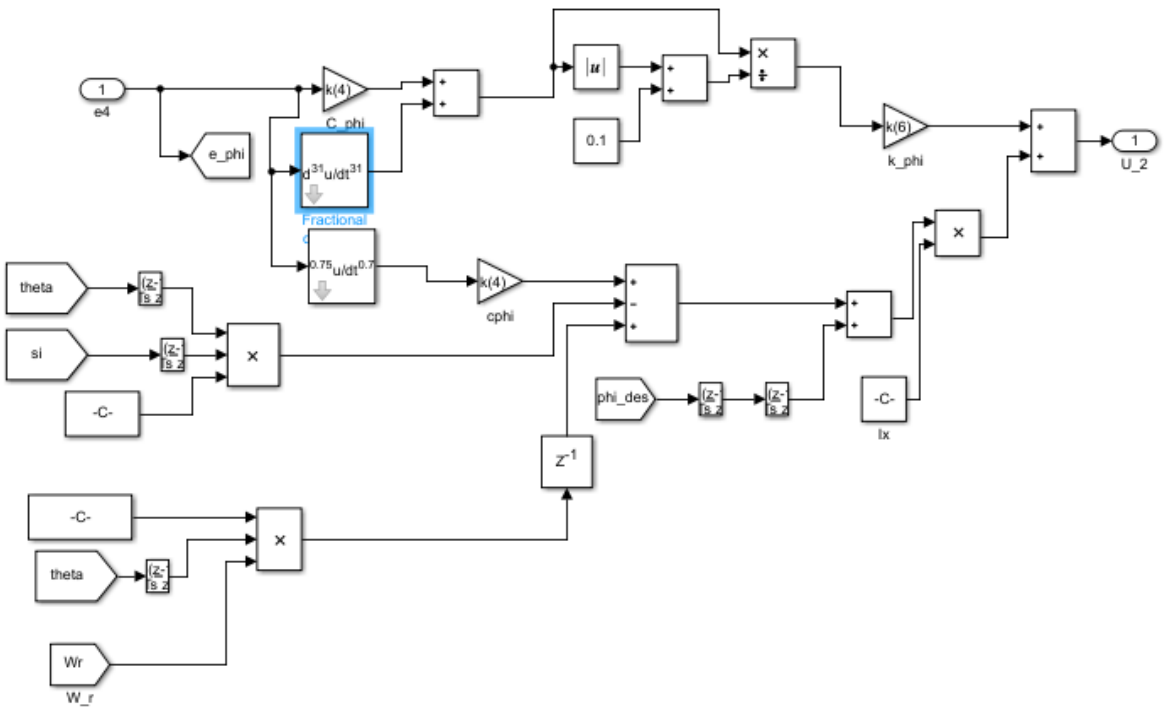


Figure (B.4) Roll movement controller extracted from block of the Complete model

B.4 Inner Loop pitch movement controller

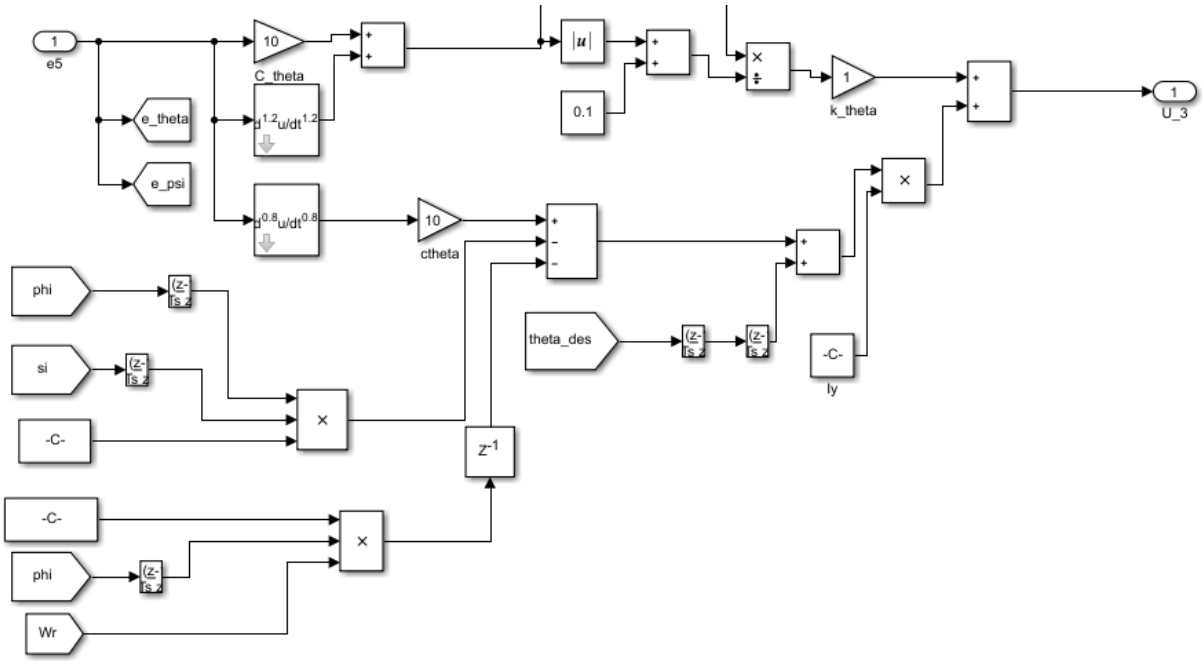


Figure (B.5) Extracting pitch controller from Figure B.2b

B.5 Inner Loop yaw movement Controller

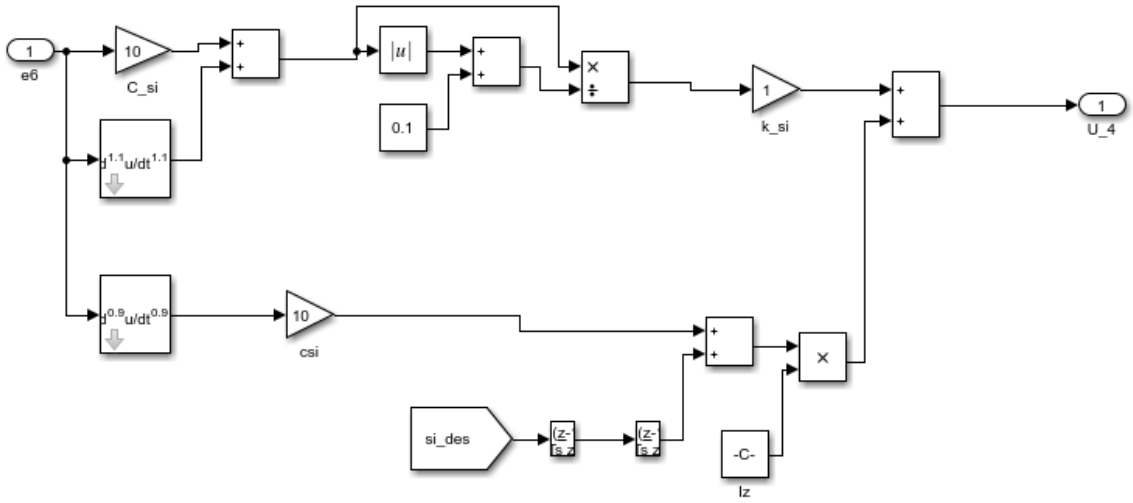


Figure (B.6) The yaw movement Controller,extracted from Figure B.2

B.6 Dynamic and kinematic model of UAV

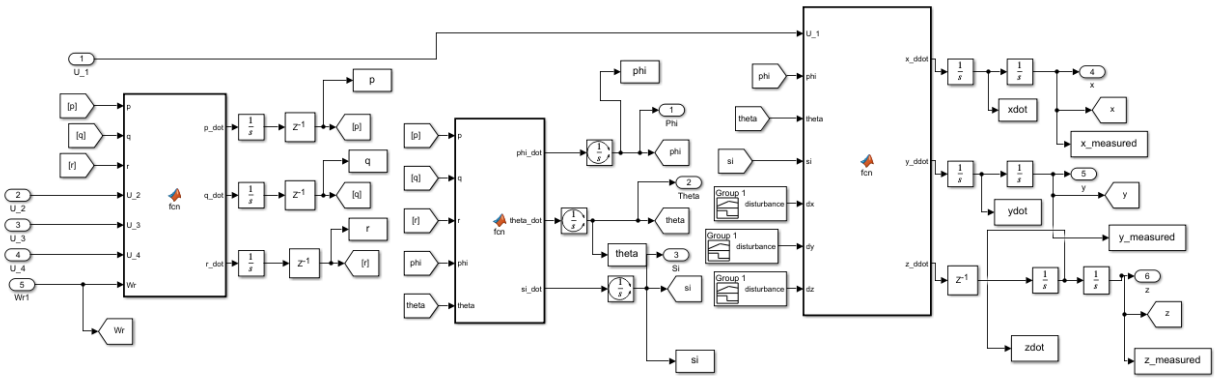


Figure (B.7) Mathematical Mode of UAV Block of Figure B.1

Appendix C

Simulink function codes

C.1 Function blocks in B.1

Listing (C.1) inverse calculation code

```
component inverse calculation
function [U_1,phi_des,theta_des] = fcn(Vx,Vy,Vz, si_des)
m=1;%mass of the quadcopter
g=9.81;%gravitational acceleration
U_1=m*sqrt(Vx^2+Vy^2+(Vz+g)^2);
theta_des=atan((cos(si_des)*Vx+sin(si_des)*Vy)/(Vz+g));
phi_des=atan(cos(theta_des)*((sin(si_des)*Vx-cos(si_des)*Vy)/(Vz+g)));
end
```

Listing (C.2) relation of control signal with speed of the propller

```
component inverse calculation
function Wr  = fcn(U1,U2,U3,U4)
b=54.2*10^-6;           % Trust coefficent

d=1.1*10^-6;              %drug coefficent

l=0.24;                   %lengthe of the arm
W_1=(sqrt((abs(0.25*(((U1/b)+((sqrt(2))*U2)/(b*l))+(((sqrt(2))*U3)/(b*l))+(U4/d))))));
W_2=sqrt((abs(0.25*(((U1/b)+((sqrt(2))*U2)/(b*l))-(((sqrt(2))*U3)/(b*l))-(U4/d)))));
W_3=(sqrt((abs(0.25*(((U1/b)-((sqrt(2))*U2)/(b*l))-(((sqrt(2))*U3)/(b*l))+(U4/d))))));
W_4=sqrt((abs(0.25*(((U1/b)-((sqrt(2))*U2)/(b*l))+(((sqrt(2))*U3)/(b*l))-(U4/d)))));
W_r=-W_1+W_2-W_3+W_4;
```

```
end
```

Listing (C.3) Euler rate on body frame

```
component Euler rate on body framen
function [p_dot,q_dot,r_dot]= fcn(p,q,r,U_2,U_3,U_4,Wr)\\
Ix=0.0081;\\
Iy=0.0081;\\
Iz=0.0142;\\
JTP=0.000104;\\
p_dot=((Iy-Iz)*q*r-(JTP*q*Wr))/Ix + U_2/Ix;\\
q_dot=((Iz-Ix)*p*r+U_3 +(JTP*p*Wr))/Iy;\\
r_dot=((Ix-Iy)*p*q)/Iz + U_4/Iz;\\
```

Listing (C.4) Euler angles from Euler rates

```
component dynamics
function [phi_dot,theta_dot,si_dot]= fcn(p,q,r,phi,theta)
phi_dot=p+q*sin(phi)*tan(theta)+r*cos(phi)*tan(theta);
theta_dot=q*cos(phi)-sin(phi)*r;
si_dot=q*sin(phi)*sec(theta)+r*cos(phi)*sec(theta);
end
```

Appendix D

Quadcopter 3d animation code

D.1 Rotation to roll pitch yaw code

Listing (D.1) rotation to roll pitch yaw code

```
component Rot2RPY_ZXY
function [phi,theta,psi] = Rot2RPY_ZXY(R)
phi = asin(R(2,3))
psi = atan2(-R(2,1)/cos(phi),R(2,2)/cos(phi));
theta = atan2(-R(1,3)/cos(phi),R(3,3)/cos(phi));
end
```

D.2 Conversion of roll pitch yaw to rotation code

Listing (D.2) roll pitch yaw to rotation code

```
component RPY2Rot
function bRi = RPY2Rot(angles)
phi = angles(1);
theta = angles(2);
psi = angles(3);
R_3 = [ cos(psi) sin(psi) 0; -sin(psi) cos(psi) 0; 0 0 1];
R_2 = [cos(theta) 0 -sin(theta); 0 1 0; sin(theta) 0 cos(theta)];
R_1 = [1 0 0; 0 cos(phi) sin(phi); 0 -sin(phi) cos(phi)];
bRi = R_1*R_2*R_3;
end
```

D.3 Input trajectory for square wave

Listing (D.3) square wave input trajectory code

```
\label{square wave input trajectory code}
component INPUT
function [x,y,z]  = fcn(t)
if t<10
x=0;
y=0;
z=3*t^2-(1/5)*t^3;
elseif (10<=t)&&(t<=20)
  x=500-120*t+9*t^2-0.2*t^3;
  y=0;
z=100;
elseif (20<=t)&&(t<=30)
  x=100;
  y=0;
z=100;
elseif (30<t)&&(t<=40)
  x=100;
  y=3240-288*t+(42/5)*t^2-(4/50)*t^3;
z=100;
elseif (40<t)&&(t<=50)
  x=100;
  y=40;
z=100;
elseif (50<t)&&(t<=60)
  x=-32400+1800*t-33*t^2+(1/5)*t^3;
  y=40;
z=100;
elseif (60<t)&&(t<=70)
  x=0;
  y=40;
z=100;
elseif (70<t)&&(t<=80)
  x=0;
  y=33360-1344*t+18*t^2-(4/50)*t^3;
z=100;
elseif (80<t)&&(t<=90)
    x=0;
    y=80;
```

```
    z=100;
    elseif (90<t)&&(t<=100)
    x=170100-5400*t+57*t^2-(1/5)*t^3;
    y=80;
    z=100;
    elseif (100<t)&&(t<=110)
    x=100;
    y=80;
    z=100;
    elseif (110<t)&&(t<=120)
    x=100;
    y=121080-3168*t+27.6*t^2-(4/50)*t^3;
    z=100;
     elseif (120<t)&&(t<=130)
    x=100;
    y=120;
    z=100;
  elseif (130<t)&&(t<=140)
    x=-490000+10920*t-81*t^2+(1/5)*t^3;
    y=120;
    z=100;
else
  x=0;
y=120;
z=100;
end
```

D.4 Main code of quadcopter animation

Listing (D.4) main code of quadcopter animation

```
component body of UAV
curve=animatedline('linewidth',2);
%shade=animatedline('linewidth',0.2*drone1_atti(3));
%set(gca,'xlim',[0 100],'ylim',[0 100], 'zlim',[0 100]);
%hold on
%view(-45,29);
drone1_initStates = [0, 0,0 , ...
% x, y, z
    0, 0, 0, ...
% dx, dy, dz
```

```
    0, 0, 0, ...
% phi, theta, psi
    0, 0, 0]';
% p, q, r
drone1_body = [ 5,      0,     0, 1; ...
                     0, -5,     0, 1; ...
               -5,      0,     0, 1; ...
                     0, 5,     0, 1; ...
                     0,      0,     0, 1; ...
                     0,      0, -3, 1]';
               % fig1 = figure('pos',[0 100 600 600]);
%h = gca;
view(3);
%fig1.CurrentAxes.ZDir = 'Reverse';
%fig1.CurrentAxes.YDir = 'Reverse';
% view([-60 30]);
% view([90 0])   % To check phi (roll attitude) only
% view([0 0])    % To check theta (pitch attitude) only
% view([270 90]) % Top view
axis equal;
hold on
plot3(out.x_measured,out.y_measured,out.z_measured,'r')
%hold on
plot3(out.xd,out.yd,out.zd,'b')
%hold on
%plot3(out.xs,out.ys,out.zs,'g')
grid on;
xlim([-30 130]);
ylim([-30 130]);
zlim([0 110]);
xlabel('X[m]');
ylabel('Y[m]');
zlabel('Height[m]');

hold(gca, 'on');

wHb = [RPY2Rot(drone1_state(7:9))' drone1_state(1:3); 0 0 0 1];
% [Rot(also contains shear, reflection, local sacling), displacement; perspective ,global s
drone1_world = wHb * drone1_body; % [4x4][4x6]
drone1_atti = drone1_world(1:3, :);
```

```
fig1_ARM13 = plot3(gca, drone1_atti(1,[1 3]), drone1_atti(2,[1 3]), drone1_atti(3,[1 3]),
        '-ro', 'MarkerSize', 5);
fig1_ARM24 = plot3(gca, drone1_atti(1,[2 4]), drone1_atti(2,[2 4]), drone1_atti(3,[2 4]),
        '-bo', 'MarkerSize', 5);
fig1_payload = plot3(gca, drone1_atti(1,[5 6]), drone1_atti(2,[5 6]), drone1_atti(3,[5 6]),
        '-k', 'Linewidth', 2);
fig1_shadow = plot3(0,0,0,'xk','Linewidth',3);
%area(fig1_shadow,'FaceColor','flat')
lx=drone1_atti(3);
ly=lx;

hold(gca, 'off');


for i=1:length(out.y_measured)


    %% 3D Plot

    wHb = [RPY2Rot(drone1_state(7:9))' drone1_state(1:3); 0 0 0 1];
    drone1_world = wHb * drone1_body;
    drone1_atti = drone1_world(1:3, :);
    set(fig1_ARM13, ...
        'XData', drone1_atti(1,[1 3]), ...
        'YData', drone1_atti(2,[1 3]), ...
        'ZData', drone1_atti(3,[1 3]));
    set(fig1_ARM24, ...
        'XData', drone1_atti(1,[2 4]), ...
        'YData', drone1_atti(2,[2 4]), ...
        'ZData', drone1_atti(3,[2 4]));
    set(fig1_payload, ...
        'XData', drone1_atti(1,[5 6]), ...
        'YData', drone1_atti(2,[5 6]), ...
        'ZData', drone1_atti(3,[5 6]));
    set(fig1_shadow, ...
        'XData', drone1_state(1), ...
        'YData', drone1_state(2));
  hold on
  F(i)=getframe(gcf);
  end
```

Appendix E

Fitness function and PSO code

E.1 PSO code

Listing (E.1) main code

```
component PSO
clc                    %to clear the command window
clear                  %to clear the work space
%%problem settings
lb=[0 0.01 0 0 00.1 0 0 0.01 0 ] ;      %lower bound
ub=[2 1 3 2 1 3 2 1 3];            %upper bound
obfun=@fosmc;  %fitness function
%%algorithem parameters
Np= 10;                  %population size
T=10;                    %no of aiteration
w=0.9;                  %ineria weight
c1=1.8;                 %acceleration cofficent
c2=1.8;                  %acceleration cofficent

%%parameter swarm opimization
f=NaN(Np,1)% vector to store fitness function value of population
bestfititer=NaN(T+1,1)
D=length(lb);%no of decition variablrs in the population
P=repmat(lb,Np,1)+repmat((ub-lb),Np,1).*rand(Np,D)%generation of initial position
v=repmat(lb,Np,1)+repmat((ub-lb),Np,1).*rand(Np,D)%generation of initial velocity
for p=1:Np
 f(p)=obfun(P(p,:))%evaluate the fitness function of the initial population
end
```

```
pbest=P %initialize the personal best position
f_pbest=f;%initialize fitness of  the personal best solution
[f_gbest,ind]=min(f_pbest);%determine the best objective fitness function value
gbest=P(ind,:);%determine the bast solution
bestfititer(1)=f_gbest;
for t=1:T
    for p=1:Np
        v(p,:)=w*v(p,:)+c1*rand(1,D).*(pbest(p,:)-P(p,:))+c2*rand(1,D).*(gbest-P(p,:));
        P(p,:)=P(p,:)+v(p,:);  %update new position
        P(p,:)=max(P(p,:),lb);%bounding violating variables to lower bound
        P(p,:)=min(P(p,:),ub);%bounding violating variables to upper bound
        f(p)=prob(P(p,:));% determine fitness of the new solution
        if f(p)<f_pbest(p)
            f_pbest(p)=f(p);%updating fitness function of personal bestsolution
            pbest(p,:)=P(p,:);%updating pbest solution

            if f_pbest(p)<f_gbest
                f_gbest=f_pbest(p);%updating fitness value of the best solution
                gbest=pbest(p,:);%updating  the best solution
            end
        end
        k
    end
    bestfititer(t+1)=f_gbest;
    disp(['iteration',num2str(t),':best fitness=',num2str(bestfititer(t+1))])
end
best_fitness=f_gbest
bestsol=gbest
```

E.2 Fitness function code

Listing (E.2) Fitness function code

```
component PSO
function cost=fosmc(k)
assignin('base','k',k);
 a = sim('slop','SimulationMode','normal');
    b = a.get('ITAE');
    cost=b(length(b));

end
```