Modeling and Control of a Pesticide Spraying Drone Using the Model Predictive Controller

Neway Yifru Adamu

A Thesis Submitted to

The Department of Electrical Power and Control Engineering

School of Electrical Engineering and Computing

Presented in Partial Fulfillment of the Requirement for the Master of Science Degree in Electrical Power and Control Engineering (Control)

Office of Graduate Studies

Adama Science and Technology University

Adama

October, 2022

Modeling and Control of @ Pesticide Spraying Drone Using the Model Predictive Controller

Neway Yifru Adamu

Advisor: Professor Gang Gyoo Jin

A Thesis Submitted to

The Department of Electrical Power and Control Engineering

School of Electrical Engineering and Computing

Presented in Partial Fulfillment of the Requirement for the Master of Science Degree in Electrical Power and Control Engineering (Control)

Office of Graduate Studies
Adama Science and Technology University

Adama

October, 2022

DECLARATION

I declare that this Master thesis entitled “Modeling and Control of Pesticide Spraying Drone Using Model Predictive Controller” is my original work. That is, it has not been submitted for the award of any academic degree, diploma, or certificate in any other university. All sources of materials that are used for this thesis have been duly acknowledged through citation

Neway Yifru Adamu	_____	_____
Name of student	Signature	Date

RECOMMENDATION

I/we, the advisor(s) of this thesis, hereby certify that I/we have read the revised version of the thesis entitled “Modeling and Control of Pesticide Spraying Drone Using Model Predictive Controller” prepared under my guidance by Neway Yifru submitted in partial fulfillment of the requirements for the degree of Mater’s of Science in Electrical Power and Control Engineering. Therefore, I/we recommend the submission of revised version of the thesis to the department following the applicable procedures.

_____	_____	_____
Advisor /Supervisor/	Signature	Date

APPROVAL PAGE

I the advisors of the thesis entitled “Modeling and Control of Pesticide Spraying Drone Using Model Predictive Controller” and developed by Neway Yifru hereby certify that the recommendation and suggestions made by the board of examiners are appropriately incorporated into the final version of the thesis.

_____	_____	_____
Advisor	signature	date

We, the undersigned, members of the Board of Examiners of the thesis by Neway Yifru have read and evaluated the thesis entitled “Modeling and Control of Pesticide Spraying Drone Using Model Predictive Controller” and examined the candidate during open defense. This is, therefore, to certify that the thesis is accepted for partial fulfillment of the requirement of the degree of Master of Science in electrical power and Control engineering.

_____	_____	_____
Chairperson	Signature	Date
_____	_____	_____
Internal examiner	Signature	Date
_____	_____	_____
External examiner	Signature	Date

Finally, approval and acceptance of the thesis is contingent upon submission of its final copy to the Office of Postgraduate Studies (OPGS) through the Department Graduate Council (DGC) and School Graduate Committee (SGC).

_____	_____	_____
Department Head	Signature	Date
_____	_____	_____
School Dean	Signature	Date
_____	_____	_____
Office of Postgraduate Studies Dean	Signature	Date

ACKNOWLEDGEMENT

First and for most, I would like to express my heartily gratitude to my supervisor, **professor Gang Gyoo Jin** for the guidance and enthusiasm given throughout the idea of this project. My appreciation also goes to my family who has been so tolerant and supports to all these years. Thanks for their inspiration, love and demonstrative supports that they had given to me.

Nevertheless, my great thankfulness dedicated to my best friends those whom involve directly or indirectly with this research. There is no such meaningful word than Thank You So Much.

TABLE OF CONTENTS

DECLARATION i

RECOMMENDATION ii

APPROVAL PAGE iii

ACKNOWLEDGEMENT iv

TABLE OF CONTENTS v

LIST OF TABLES ix

LIST OF FIGURES x

ACRONYMS xii

LIST OF SYMBOLS xiv

ABSTRACT xv

CHAPTER ONE 1

1. INTRODUCTION 1

1.1. Background 1

1.2. Problem of the statement 3

1.3. Objective of the study 4

1.3.1. General Objective 4

1.3.2. Specific Objective 4

1.4. Scope of the Study 4

1.5. Limitation of the Study 4

1.6. Motivation of the Study 4

1.7. Significance of the Study 5

1.8. Thesis Organization 5

CHAPTER TWO 6

2. LITERATURE REVIEW AND BASIC THEORIES 6
2.1. Chapter Overview 6
2.2. Unmanned Aerial Vehicles (UAVs) 6
2.3. History of UAVs 6
2.4. Applications of UAVs 7
2.5. Classification of UAVs 8
2.6. Quadcopters 8
2.7. Related Works 9
CHAPTER THREE 12
3. MATERIALS AND METHODS 12
3.1. Chapter Overview 12
3.2. Materials 12
3.3. Methods 12
3.4. The Proposed Block Diagram for quadcopter 13
3.5. Coordinate Frames 14
3.6. Quadcopter Dynamics 17
3.6.1. Actuator Dynamics 17
3.6.2. Translational equation of motion 18
3.6.3. Rotational Equation of Motion 19
3.6.4. State Space model 20
3.7. Controllability and observability 24
3.7.1. Controllability 24
3.7.2. Observability 24
CHAPTER FOUR 25
4. ESTIMATOR AND CONTROLLER DESIGN 25

4.1. Chapter Overview 25
4.2 Estate Estimation 25
4.2.1. Kalman Filter (KF) 25
4.2. Model Predictive Control 28
4.2.1. Output Prediction 29
4.2.2. Constraints 31
4.2.3. Optimization 33
4.2.4. MPC Mathematical Formulation 35
CHAPTER FIVE 37
5. RESULT AND ANALYSIS 37
5.1. Proposed Control System Result 38
5.2. Trajectory Tracking Performance without Disturbance 39
5.2.1. Roll rate controller tracking performance 39
5.2.2. Pitch rate controller tracking performance 40
5.2.3. Yaw rate controller tracking performance 41
5.3. Trajectory Tracking Performance with Disturbance 42
5.3.1. Roll rate controller tracking performance 43
5.3.2. Pitch rate controller tracking performance 44
5.3.3. yaw rate controller tracking performance 45
5.4. Trajectory tracking performance with estimator 46
5.4.1. Position tracking performance without Kalman filter 46
5.4.2. Position tracking performance with Kalman filter estimator 47
5.5. Software in the loop gazebo simulation result 48
CHAPTER SIX 50
6. CONCLUSION AND RECOMMENDATION 50

6.1. Conclusion 50
6.2. Recommendations 50
7. REFERENCE 52
APPENDICES 56
A. MATLAB script for MPC design 56
B. MATLAB script for MPC predictions output 61
C. Controllability, observability and stability 62

LIST OF TABLES

Table 5. 1: Motor Specification and rating 37
Table 5. 2: Physical parameter for quadcopter 37
Table 5. 3: MATLAB MPC simulation parameters for $n_u = 3$, $n_y = 6$..... 38
Table 5. 4: MATLAB MPC simulation parameters for $n_u = 3$, $n_y = 6$, with disturbance 42

LIST OF FIGURES

Figure 1.1: Quadrotor Configuration 1
Figure 1.2: pesticide Spraying drone 2
Figure 2.1: Oehmichen No.2 Quadrotor 6
Figure 2.2: The de Bothezat Quadrotor 7
Figure 2.3: + and X type quadcopter (left to right) 8
Figure 3.1: Work flow 13
Figure 3.2: Block diagram of the proposed quadcopter control system 14
Figure 3.3: Earth-fixed and body-fixed coordinate frames 14
Figure 3.4: Body- fixed frame in NEU frame 15
Figure 3.5: The forces & moments generated by the motors. 18
Figure 4.1: State estimation and controller 25
Figure 4.2: Steps of MPC 29
Figure 4.3: Prediction Horizon 29
Figure 4.4: Prediction and optimization 35
Figure 5. 1: roll angle angular rate trajectory 39
Figure 5. 2: Roll rate controller tracking error 40
Figure 5. 3: Pitch rate reference trajectory tracking controller using MPC 41
Figure 5. 4: pitch rate controller tracking error 41
Figure 5. 5:Yaw rate reference trajectory controller performance using MPC 42
Figure 5. 6: Yaw rate controller tracking error 42
Figure 5. 7: MATLAB roll rates 43
Figure 5. 8: Roll rate controller tracking error 44
Figure 5. 9: MATLAB pitch rates 44
Figure 5. 10: pitch rate controller tracking error 45
Figure 5. 11: MATLAB yaw rates 45
Figure 5. 12: Yaw rate controller tracking error 46
Figure 5. 13: MATLAB position control without Kalman 47
Figure 5. 14: MATLAB position control without Kalman 47
Figure 5. 15: Gazebo flight simulation during flight with z= 2m, hovering 48

Figure 5. 16: roll angle from SITL simulation 48
Figure 5. 17: pitch angle from SITL simulation 49

ACRONYMS

AFC	Automatic flight Controller
BLDC	Brushless DC Motor
CAD	Computer Aided Design
DC	Direct Current
ESC	Electronic Speed Controller
FPV	First Person View
GNSS	Global Navigation Satellite System
GPIO	General Purpose Input/output
GPS	Global Positioning System
IMU	Inertial Measurement Unit
KV	Constant Velocity (of a motor)
LQ	Linear Quadratic
NLP	Nonlinear Programming problem
PID	Proportional Integral Derivative
PWM	Pulse Width Modulation
RC	Radio Control
RGB	Red Green Blue
RPM	Revolutions Per Minute
RTOS	Real Time Operating System
ROS	Robot operating system
UART	Universal Asynchronous Receiver-Transmitter

UAV	Unmanned Air Vehicle
UDP	User Datagram Protocol
VTOL	Vertical Takeoff and Landing

LIST OF SYMBOLS

d	Aerodynamic drag contribution
B	Body reference frame
E	Earth reference frame
F_{ext}	Sum of external forces acting on the quadrotor
Fi	Force generated by i^{th} rotor
g	Gravitational acceleration constant
$Ixx; Iyy; Izz$	inertia
Jr	Rotor inertia
l	Length of the moment arm
m	Mass of quadrotor
M_{ext}	Sum of external moments acting on the quadrotor
Mi	Moment exerted on the quadrotor by i^{th} rotor
R	Transformation matrix
RBE	Transformation matrix from FE to FB
REB	Transformation matrix from FB to FE
Rr	Angular transformation matrix from FE to FB
Ti	Torque generated by ith rotor
$U1; U2; U3; U4$	Control inputs
VB	Translational velocity vector of quadrotor
$p; q; r$	Body angular velocities
$u; v; w$	Body velocities
x	Position Vector of Quadrotor
t	Motor time constant

ABSTRACT

One of the primary sources of income in Ethiopia's economy is Agriculture. Over 80% of the people depend upon the agriculture fields. Agriculture suffers significant losses due to diseases spread by pests and insects, which reduce crop productivity. In order to improve crop quality, pesticides and fertilizers are used to kill insects and pests. However, almost all agricultural fields in Africa, particularly in Ethiopia, use humans to apply pesticides and fertilizers to the entire crop field, resulting in pesticide waste and adverse effects on human health. Unmanned Aerial Vehicles (UAVs) are becoming increasingly popular as agricultural research topics. The purpose of this research is to control the attitude and position of an autonomous pesticide spraying quadcopter drone using a model predictive controller (MPC). Localization, mapping, and trajectory following of a quadrotor are achieved using data from a camera, IMU (Inertial Measurement Unit), GPS (Global Positioning System), and range finder in a disturbance-based environment such as wind and obstacles. In this study, the quadcopter model is modeled using the newton Euler formulation of motion to implement this controller. The Kalman filter (KF) is used to estimate the states of the quadcopter and the variation in quadcopter mass due to spraying. The controller was implemented in the quadcopter using MATLAB to test the feasibility of the proposed control method. The simulation shows that the proposed controller adapts to both setpoint and trajectory tracking. The PX4 firmware includes a Proportional Integral Derivative (PID) controller. MPC, on the other hand, has never been used on commercial drones because it requires a mathematical model of the specific quadcopter to be used. Thus, the purpose of this thesis is to assess the practicability of the MPC control scheme for quadcopters with the modified PX4 firmware to control agricultural drones. The MATLAB/Simulink simulation shows that the proposed controller rejects the disturbance and state noise based on estimator from Kalman filter to the unknown system noise of quadcopter and it performs effectively at the reference of altitude, position and attitude.

Key Words: - *Model Predictive Controller (MPC), Extended Kalman Filter (EKF), Moving Horizon Estimation (MHE), Proportional-Integral-Derivative (PID), Robot operating system (ROS), Range Finder, PX4*

CHAPTER ONE

1. INTRODUCTION

1.1. Background

Unmanned Aerial Vehicles (UAVs) in general, and multirotor aircraft in particular, have recently received a lot of attention due to their high versatility and simple mechanical architecture. Quadcopters or quadrotors are UAVs that have four rotors. Recently, the quadcopter has emerged as one of the most widely researched control systems. It serves as an excellent model for studying the behavior of multiple-input multiple-output (MIMO) and nonlinear systems. The quadcopter is an under-actuated system with 6dof (three translational and three rotational) (Fernando et al., n.d.), but with only four independent inputs (the speed of each motor), and with only four independent inputs (rotor speed), this results in the coupling of rotational and translational dynamics. One of the major challenges with these flying vehicles has been to make them self-sufficient, requiring little to no human effort to fly. To deploy these systems in real-world scenarios, we need robust answers to the autonomy system's components, specifically state estimation, control, mapping, and planning. This autonomous flight capability can enable a variety of applications such as disaster response, infrastructure inspection, agricultural monitoring (Khan et al., 2021) and pesticide spraying, which currently requires significant direct human engagement and effort. The quad-copter, as shown in Fig.1.1, is made up of two pairs of counter-rotating rotors located at the ends of a cross-frame that is symmetric about the center of gravity, which matches with the origin of the reference system. Each rotor has a propeller attached to a separately powered BLDC motor.

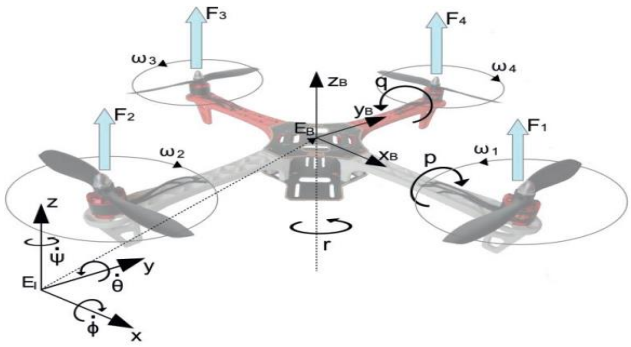

Figure 1.1: Quadrotor Configuration

The rapid expansion of the human population elevates the fundamental need for high productivity, high performance, and sustainable farming (Mogili & Deepak, 2018). In the present world, agriculture is necessary for the survival of more than 60% of the world's population (Khan et al., 2021). It is a critical factor in the conservation of the environment in the developing world. Agriculture must be modernized in order to increase production and food supply. UAVs are among the most useful machines in modern agriculture in which used to Spray pesticides and fertilizer in agricultural fields (Chavan, 2019). Precision farming, crop monitoring, geographical tracking, and field mapping are just a few sectors of agricultural drone applications (Mogili & Deepak, 2018) and (Khan et al., 2021).

Figure 1.2: pesticide Spraying drone

Different mathematical techniques can be used to derive a quadcopter's dynamic equations. The Newton-Euler method and the Euler-Lagrange method are two commonly used methods. The Newton-Euler method is based on Newton's second law of motion and focuses on rigid body equations (Njinwoua & Wouwer, 2018). To obtain the mathematical model, the Euler-Lagrange method employs kinematics and energy equations (Ordaz et al., 2018). Ideally, both techniques should yield the same mathematical model. The Newton-Euler method was used in this paper. The obtained equations can be represented in state-space form. As an ASTU Space Technology Institute (STI) member, we are currently working on quadcopter drone research and experimental development using an Arduino microcontroller and an IMU sensor.

This thesis work presents learning-based Model predictive approaches for controlling a quadcopter's position and attitude. The system dynamics have been used to design linear and nonlinear mathematical models. To adapt the system parameters, a Kalman filter estimates the state and changes the parameter. At each moment, the estimator sends to the controller a set of estimated plant parameters computed based on the previous plant input and output and system dynamic model, and then a control input based on the optimal control approach is calculated. This control input generates a new plant output, and the entire sequence of parameters and input updates is repeated.

1.2. Problem of the statement

Agriculture is one of Ethiopia's main sources of income; approximately 80 percent of Ethiopians depend on agriculture, primarily subsistence and rain-fed farming and animal agriculture. Despite this, approximately 35% of the population lacks access to healthy food. Furthermore, the agriculture field suffers significant losses due to diseases transmitted by pests and insects, (MoARD(Ministry of Agriculture and Rural Development), 2010) which minimize agricultural production. The agricultural drone is one solution for spraying pesticides; nevertheless, the agricultural drone's mass changed while spraying pesticides; this Challenge introduces additional uncertainties to the system, which can be modeled as a variation. As a result, system performance and control input for a model-free control approach become unstable. To address this issue, the model-based controller is guaranteed an appropriate mathematical model of the system. As a result, for guaranteed effectiveness from a model-based controller, the aforementioned variations must be learned over time, and the controller must adapt to changing conditions.

Therefore, the goal of this study is to design and control pesticide spraying quadcopter drones using a learning-based model predictive controller with uncertain and/or time-varying dynamic models.

1.3. Objective of the study

1.3.1. General Objective

The general objective of this research work is to model and control pesticide spraying agricultural drone using MPC controller.

1.3.2. Specific Objective

- ✓ To investigate the nonlinear model of a quadcopter drone
- ✓ To linearize and formulate the system in the state space form
- ✓ To design a model predictive controller (MPC) for position and attitude control
- ✓ To simulate the 3D physical simulation of the quadcopter using Gazebo
- ✓ To implement the proposed control method and simulate its performance with a software-in the loop (SITL) simulator

1.4. Scope of the Study

The scope of this thesis is to experimentally implement autonomous pesticide spraying drone for agriculture. The thesis is implemented using MATLAB and SITL implementation using ROS. The implementation is using KF in PX4 firmware in SITL.

1.5. Limitation of the Study

The existing agricultural drone uses 15-20 letters of a pesticide storage tank and has a payload of more than 20 kg, but due to battery power constraints in the country and motor limitations, the implementation is limited to a 1.5 kg overall load as well as a light-weighted frame made of the aluminum plate was used to reduce the weight of the drone. The data was collected using the onboard IMU from the Indore flight to reduce the risk of wind disturbance so that flights could be conducted with the necessary precautions.

1.6. Motivation of the Study

The development and initiation of exciting development in quadcopter drones have created new ways of applications for it. This type of vehicle can navigate in tight spaces and is used to complete a variety of tasks. Drones are now used for a variety of purposes in a variety of fields. As a result, the design of control algorithms is critical in the development of these technologically advanced systems.

1.7. Significance of the Study

Controlling unmanned aerial vehicle's (UAV) position, altitude, and orientation is an exciting area of control system research. Because of the multifunctional application areas and advantages of quadrotor UAVs, these types of studies are currently being conducted in a variety of locations. Ethiopia is new to this type of technology and innovation, so these types of studies are beneficial in creating a technologically oriented generation. The Space Technology Institute (STI) Center of Excellence at Adama Science and Technology University (ASTU) is an example of technological excellency for developing research in the country. The STI development teams are working to create quadcopter drones for the Ethiopian market.

1.8. Thesis Organization

The thesis is organized as follows:

Chapter 1: Introduces the historical background of UAVs, including a statement of the problem, the objective of the thesis, scope, limitation, motivation, and significance of the thesis.

Chapter 2: deals with the literature works previously done in the area of quadcopter control and a qualitative introduction on the principles of working of a quadrotor is discussed

Chapter 3: The mathematical model is presented based on Newton-Euler formalism and a linearized version of the model is obtained Based on this model using equilibrium points.

Chapter 4: In this chapter the methodology of designing and implementing model predictive control strategy is covered. KF is designed for linearized and nonlinear dynamic model of the quadcopter.

Chapter 5: The simulations obtained from running the MPC controller in MATLAB, SITL Gazebo, and the state estimation are presented and discussed.

Chapter 6: Concludes the thesis with a discussion of the significance of the results obtained, challenges faced and recommendations for future work.

CHAPTER TWO

2. LITERATURE REVIEW AND BASIC THEORIES

2.1. Chapter Overview

This chapter provides a brief description, history, types, and applications of the UAV quadcopter. Following that, the notion and architecture of quadcopter-type UAVs are discussed. Finally, closely related works are reviewed and discussed, together with their benefits and drawbacks.

2.2. Unmanned Aerial Vehicles (UAVs)

An unmanned aerial vehicle (UAV), usually known as a drone, is an aircraft without any human pilot, or customers on board (Praveenkumar & Prabhakaram, 2014). The flight of UAVs may work under remote control by a human pilot, as remotely-piloted airplane (RPA), or with various degrees of self-sufficiency, such as autopilot assistance, up to fully autonomous aircraft that have no provision for human interference. UAVs were mainly used in military (Chavan, 2019) application but recently they are being deployed in civil applications.

2.3. History of UAVs

Etienne Oehmichen was the first researcher who investigated rotorcraft designs in 1920. Among the six designs he tried, his second multirotor had four rotors and eight propellers, all energized by a single engine. The researcher used a steel-tube frame (Venkatesh et al., 2017), with two-bladed rotors at the ends of the four arms. The angle of these blades could be diverse by bending.

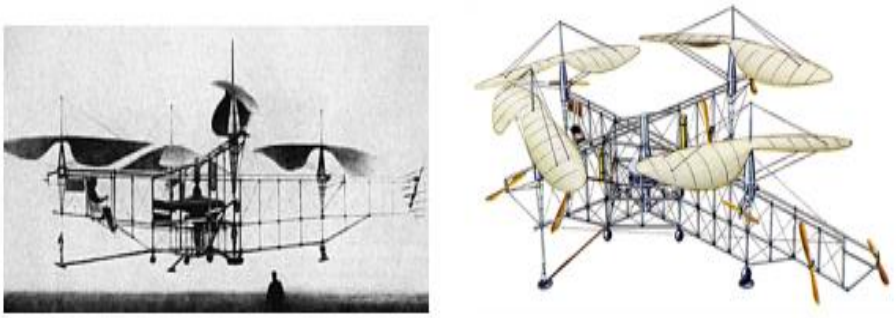

Figure 2.1: Oehmichen No.2 Quadrotor

After Oehmichen, Dr. de Bothezat and Ivan Jerome technologically advanced this aircraft, with six bladed rotors at the end of an X-shaped structure. Two small propellers with adjustable pitch were used for thrust and yaw control. The vehicle used joint pitch control. Its initial flight in October 1922. About 100 flights were complete by the end of 1923. The maximum it eternally reached was about 5m.

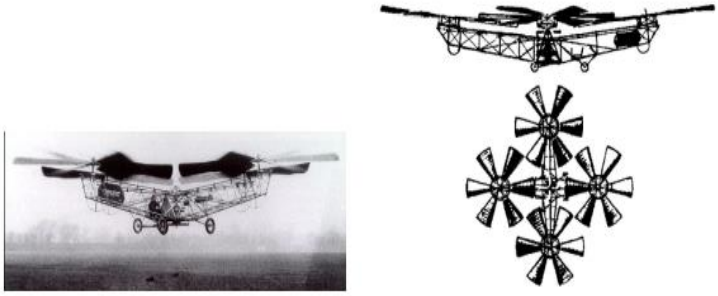

Figure 2.2: The de Bothezat Quadrotor

2.4. Applications of UAVs

UAVs were mainly used in military application but recently they are being deployed in civil applications too [5]. In addition to the military use, UAVs can be used in many civil or commercial applications that are too hard, too complex or too dangerous for manned aircrafts. These are

- **Earth Science:** Measuring deformations in the Earth's crust that may be indications to natural disasters like earthquakes, landslides or volcanos.
- **Search and rescue:** UAVs equipped with cameras are used to search for survivors after natural disasters like earthquakes and survivors from shipwrecks and aircraft crashes [9, 10].
- **Border surveillance:** UAVs are used to patrol borders for any intruders, illegal immigrants or drug and weapon smuggling.
- **Research:** UAVs are also used in research conducted in universities to proof certain theories.
- **agricultural:** UAVs are used in various agricultural sector to monitor crop, inspect and spray pesticides and fertilizers.

2.5. Classification of UAVs

The UAV can be generally classified in two categories i.e., autonomous aero-plane and autonomous helicopter. The helicopters have clear advantages over the aero-planes due to their specific capabilities like improved hovering operation, ability of landing/take-off in limited space (VTOL), etc. Unmanned Aerial Vehicles (UAVs) are either controlled remotely by a radio transmitter or from a computer ground station.

2.6. Quadcopters

A quadcopter is a multirotor (helicopter) Unmanned Aerial Vehicle (UAV) with four rotors, which are varied in speed to change the altitude and/or angular position of the vehicle. Quadcopters are configured with propellers that are connected either in an "X" or "+" configuration each propeller is powered by an electric motor (Giernacki et al., 2020). The propellers are connected in counter-rotating pairs in order to cancel out the torques created by a set of propellers rotating in one direction. In figure 2.3 (Right), the propellers labeled 1 and 3 rotate in a counter-clockwise direction, while the propellers labeled 2 and 4 rotate in a clockwise direction.

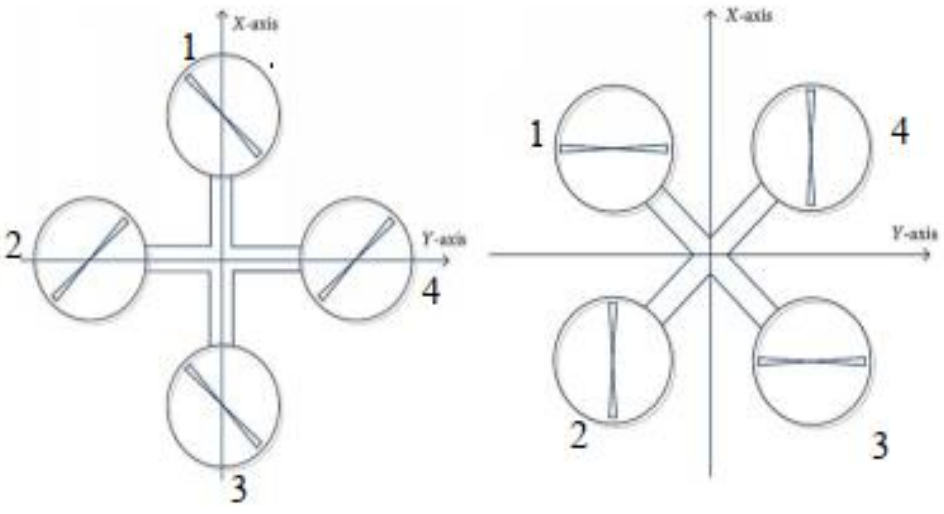

Figure 2.3: + and X type quadcopter (left to right)

2.7. Related Works

The UAV Technology required extensive research into comparable systems. By reviewing others work, we used this insight to develop our system. To this end, research papers from various groups were used as guides in the development of the dynamics and control theory for quadcopter.

A comparative analysis of Adaptive PID, SMC, PID control system approaches for quadrotor UAV attitude and position, stabilization has been formulated in (Noordin et al., 2021) The authors implemented the proposed system based on external disturbances like the aerodynamic effect.

Model predictive controller (MPC) has been used in (Dydek et al., 2013) to reject the wind disturbances that a quadrotor faces during the flight. The designed controller has used the piecewise model of the quadrotor dynamics, in which turbulence affects has been included as a disturbance. The authors verified their algorithm with experimental flights, and have showed the capability of quadrotor's attitude tracking while subjected to the wind disturbances.

The implementation of UAVs for crop monitoring and pesticide spraying is briefly reviewed in (Mogili & Deepak, 2018). This paper explains about the different commercial control module, sensors, actuator and sprinkling system used for agricultural drone and the limitation of human pesticide spraying technique.

In (Khan et al., 2021) Application of image recognition for recognized sprayer is developed using deep learning by the authors. A two-step target recognition system was developed by using deep learning for the images collected from a UAV. Agriculture cropland of coriander was considered for building a classifier for recognizing spraying areas.

In (Cárdenas R et al., 2020) the authors explain Mathematical Modelling and Identification of a Quadrotor. The authors implemented the proposed system neglecting Gyroscopic and some aerodynamics effects, as a result the simplified model using PID controller does not have good performance.

Attitude control of a cascade sliding mode control (CSMC) for quadrotor dynamics has been carried out in (Rq et al., n.d.). The proposed control system was implemented for the roll, pitch,

and yaw control by considering external disturbances to the system. As a result, the CSMC has a good result for constant mass parameter.

In (Hystad, 2015), the author developed and model a quadcopter implemented on an Arduino microcontroller in Norwegian University of Science and Technology in his master. The paper is very useful based on the hardware requirement ad motivation for prototyping. This paper aims to use understandable system descriptions and sensor models as a basis to design configurable estimators and controllers, and to build a quadcopter well suited for educational purposes.

In (Lee & Kang, 2021) authors proposed a data-based control technique based on a deep neural network (DNN). The DNN is trained with closed-loop driving data of an NMPC. The proposed "DNN control technique based on NMPC driving data" achieves control characteristics comparable to those of a well-tuned NMPC within a reasonable computation period, which is verified with an experimental scaled-car platform and realistic numerical simulations. This paper work is for unmanned ground vehicle (UGV).

In (Fernando et al., n.d.) Modeling, Control and Simulation of Quadrotor UAV has been discussed. In this paper model of quadrotor has been developed considering Newton Euler formalism, and then a PID correction was tuned to assure the desired performance after considering first order linearization for the hovering mode. Simulation was manipulated within MATLAB® Simulink environment, for which results can be observable for many controlling approaches and with variety of parameters ranging in order to get a better design perspective.

In (Fessi & Bouallègue, 2016) Modeling and Optimal LQG Controller Design for a Quadrotor UAV is designed. In this paper all aerodynamic forces and moments of the Quadrotor UAV are described within an inertial frame and a dynamical model is obtained thanks to the Newton-Euler formalism. An optimal LQG controller is then designed for the attitude and altitude stabilization of the plant, linearized around an equilibrium flight point. Several simulation results are carried out in order to show the effectiveness and robustness of the proposed LQG-based flight stabilization approach.

In (Abeywardena et al., 2013) Improved State Estimation in Quadrotor MAVs: A Novel Drift-Free Velocity Estimator is designed. In this work Dynamic equations which relate acceleration, attitude and the aero-dynamic propeller drag are encapsulated in an extended Kalman filter

framework for estimating the velocity and the attitude of the quadrotor. It is demonstrated that exploiting the relationship between the body frame accelerations and velocities, due to blade flapping, enables drift free estimation of lateral and longitudinal components of body frame translational velocity along with improvements to roll and pitch components of body attitude estimations.

In (Argentim et al., n.d.) PID, LQR and LQR-PID on a Quadcopter Platform is discussed. These paper aims to present a comparison between different controllers to be used in a dynamic model of a quadcopter platform. The controllers assumed in this work are an ITAE tuned PID, a classic LQR controller and a PID tuned with an LQR loop. The results were obtained through simulations for 10 different attitudes of the quadcopter, however, in this paper simulation results had been presented for the vertical attitude only.

Table 1 summarizes the comparison of the various algorithms as applied to quadrotors from the above literatures

No	Author and publisher	Control technique	Strength	Limitation
1	(Noordin et al., 2021), Springer	Adaptive PID, SMC, PID	Improved performance based on the comparison criteria of fitness function IAE than others	parameter uncertainty like variable mass is not considered
2	(Cárdenas R et al., 2020), Springer	Classical PID	Nonlinear model provided, thrust & drag coefficients as well as motor inertia are involved	Gyroscopic & aerodynamic effects neglected, no state estimation
3	(Lee & Kang, 2021), Springer	NMPC, DNN	effective and more robust for an optimized control	requires large computation power and computation time,
4	(Argentim et al., n.d.), IEEE, 2013	PID, LQR & LQR-PID	robust, versatile and easy implementable controller	Lack of state ad parameter Constraint

CHAPTER THREE

3. MATERIALS AND METHODS

3.1. Chapter Overview

In this chapter, the mathematical model of the quadcopter is formulated using Newton-Euler approach. The derived mathematical model of the quadcopter is linearized using Taylor linearization formulation and represented in state-space form and the controllability and observability of the dynamic system is briefly explained.

3.2. Materials

In this research work, software such as MATLAB/2021a, Gazebo, QGroundControl, Microsoft office 2019, Visual Studio Code 1.6.7, and Math Type 7.4.8.0 is used. MATLAB is the calculating software that contains technical toolboxes and Simulink. Gazebo is an open-source 3D robotics physical simulator and It combined the ODE physics engine and support code for sensor simulation and actuator control. Microsoft office is used for editing the documents. Visual Studio Code 1.6.7 is code editor made by Microsoft for Windows, Linux and macOS and provision for debugging, syntax highlighting, intelligent code completion, and code refactoring. Math Type 7.4.8.0 version is used for writing mathematical formulas and equations in Microsoft office.

3.3. Methods

The method used for this research study to accomplish the required task is shown in Figure 3.1. The most related papers and literature are reviewed and data are collected. After analyzing all of the resources, the next step is to complete the mathematical modeling for the quadcopter, which

is the foundation of this thesis. Depending on the mathematical model, the effective controller which is model based controller is selected and for getting best estimated data and parameter the state and estimator is designed using Kalman filter (KF). To evaluate the system's performance, MATLAB, and Gazebo is used as simulation tools. The block diagram in Figure 3.1 summarizes the method followed throughout the research work.

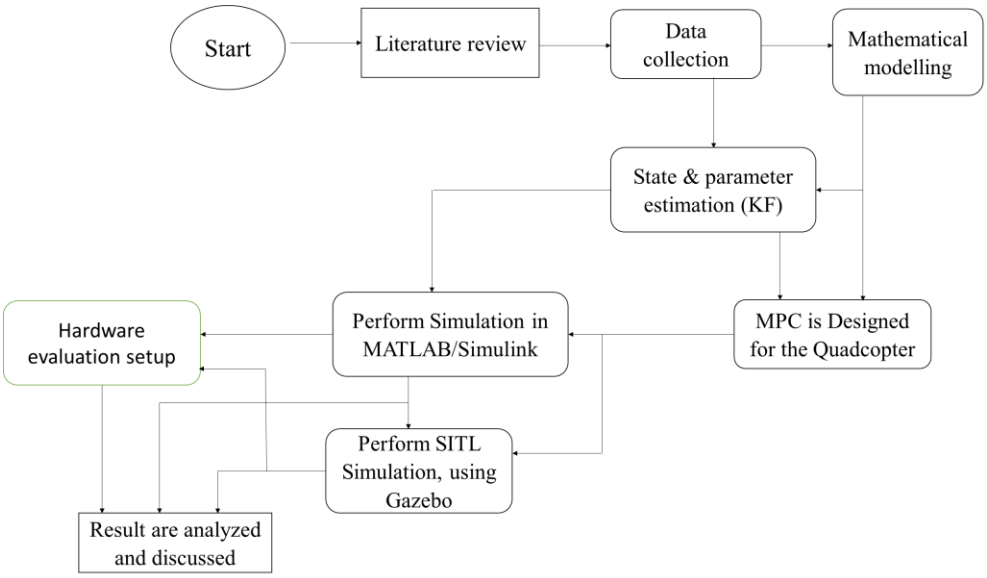

Figure 3.1: Work flow

3.4. The Proposed Block Diagram for quadcopter

Figure 3.2 shows a design of model predictive controller for controlling the quadcopter's position and orientation. Before the beginning of each optimization loop, the MPC block enhances the position and orientation with the quadcopter state-space model. This is done in order to obtain more precise quadcopter state predictions to be used in the optimizer.

The optimizer block receives position and rate setpoints. The cost function and system constraints are also provided to the optimizer. In the optimizer block, the nonlinear programming (NLP) or Quadratic Programming (QP) procedure is used to minimize the cost function while keeping predefined quadcopter constraints in mind. The MPC controller's result is the control or input vector.

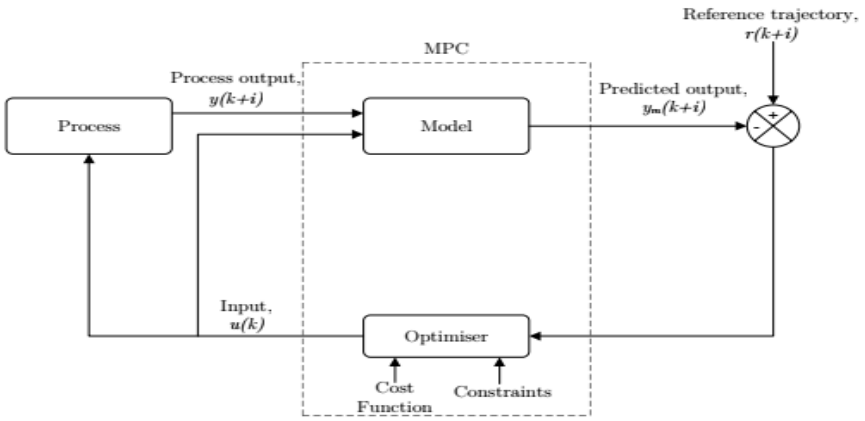

Figure 3.2: Block diagram of the proposed quadcopter control system

3.5. Coordinate Frames

Before the quadcopter can be mathematically modeled, coordinate frames must be used to explain its movement. To accomplish this, two coordinate frames are used: the inertial (Earth-fixed) frame, E_I, and the body-fixed frame, E_B. Variables measured or observed in the earth-fixed frame can be related to the body-fixed frame using two coordinate frames, and vice versa. Figure 3.3 represents these coordinate frames with the quadcopter in the body-fixed frame. The earth-fixed frame is taken as the reference frame using the NEU (North East Up) convention where the x-axis of the frame is the north, y-axis pointed to the west and the z-axis pointed up (Carrillo et al., 2014).

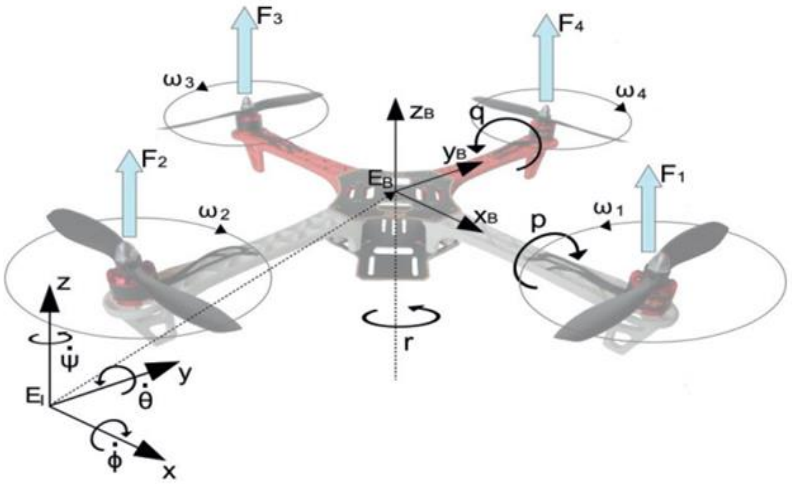

Figure 3.3: Earth-fixed and body-fixed coordinate frames

The body-fixed frame, ***B***, has its origin at the center of mass of the quadcopter. The orientation of the quadcopter, known as its **attitude**, is expressed in the body-fixed frame by Euler angles

ϕ, θ, ψ which correspond to the roll, pitch and yaw angles. The angles are also referred to as the angular positions of the quadcopter. In this paper, the inertial frame uses NEU (x-axis points to the north, the y-axis to the west and the z-axis upward) coordinates.

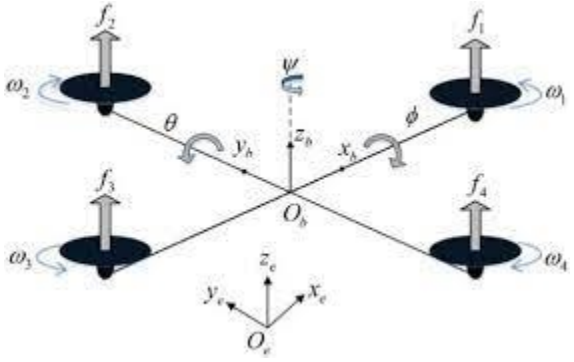

Figure 3.4: Body- fixed frame in NEU frame

In order to relate the orientation of the quadcopter in the inertial frame, rotation matrix, ***R***, is required. Assuming the order of rotation to be roll (ϕ), pitch (θ) then yaw (ψ), the rotation matrix, ***R*** which is derived based on the sequence of principle rotations is:

$$R_{x,\varphi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi & \cos\varphi \end{bmatrix} \tag{3.1}$$

$$R_{y,\theta} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{3.2}$$

$$R_{z,\phi} = \begin{bmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

To be able to calculate the total conversion from the body frame to the navigation frame (R_{IB}) all the 3 matrixes can be multiplied together. The result is a complete rotation matrix of Equation 3.1 to 3.3 (Cowling et al., n.d.) which is from the body-fixed frame to the inertial frame is given as:

$$R_{zyx}(\phi,\theta,\psi)_{IB} = \begin{bmatrix} c(\theta)c(\psi) & s(\phi)s(\theta)c(\psi) - c(\phi)s(\psi) & c(\phi)s(\theta)c(\psi) + s(\phi)s(\psi) \\ c(\theta)s(\psi) & s(\phi)s(\theta)s(\psi) + c(\phi)c(\psi) & c(\phi)s(\theta)s(\psi) - s(\phi)c(\psi) \\ -s(\theta) & s(\phi)c(\theta) & c(\phi)c(\theta) \end{bmatrix} \tag{3.4}$$

Where C(∗) = cos(∗), S(∗) = sin(∗) and T(∗) = tan(∗). The calculated matrix gives the convention from BODY to Inertia frame. In some cases, it can be useful to transform from Inertia frame to BODY frame.

The IMU used to obtain the angular velocity of the quadcopter in the body-fixed frame. A transformation is needed to relate Euler rates, $\dot{\eta} = \begin{bmatrix} \dot{\phi} & \dot{\theta} & \dot{\psi} \end{bmatrix}^T$, that are measured in the inertia frame and the angular body-fixed rates, $\omega = \begin{bmatrix} p & q & r \end{bmatrix}^T$, The transformation is obtained from (Luukkonen, 2012) and is as follows,

$$\begin{aligned} \omega = \begin{bmatrix} p & q & r \end{bmatrix}^T &= \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + R_x(\phi) \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + R_x(\phi) R_y(\theta) \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix}, \\ \begin{bmatrix} p \\ q \\ r \end{bmatrix} &= \begin{bmatrix} 1 & 0 & -s(\theta) \\ 0 & c(\phi) & s(\phi)c(\theta) \\ 0 & -s(\phi) & c(\phi)c(\theta) \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \end{aligned} \tag{3.5}$$

Since the values of p, q and r, are obtained through onboard sensors on the quadcopter, Euler angle rates are computed using the inverse of above equation

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & s(\phi)t(\theta) & c(\phi)t(\theta) \\ 0 & c(\phi) & -s(\phi) \\ 0 & \dfrac{s(\phi)}{c(\theta)} & \dfrac{c(\phi)}{c(\theta)} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{3.6}$$

Generally, from three-dimension body dynamics, it follows that the two reference frames (inertial and body) are linked by the following relations:

$$\begin{cases} \dot{\eta} = W_\eta . \omega \\ \omega = W_\eta^{-1} \dot{\eta} \\ V_I = R_{IB} . V_B \end{cases}, W_\eta = \begin{bmatrix} 1 & s(\phi)t(\theta) & c(\phi)t(\theta) \\ 0 & c(\phi) & -s(\phi) \\ 0 & s(\phi)/c(\theta) & c(\phi)/c(\theta) \end{bmatrix} \tag{3.7}$$

Where $V_I = \begin{bmatrix} \dot{x} & \dot{y} & \dot{z} \end{bmatrix}^T$, V_I and V_B translational velocity in inertial and body frame.
$V_B = \begin{bmatrix} u & v & w \end{bmatrix}^T$

Taking small angle assumptions, the transformation matrix can be reduced to an identity matrix and the body rate and Euler rate become equal (Nagaty et al., 2013). Therefore,

$$\dot{\eta} = \omega \tag{3.8}$$

3.6. Quadcopter Dynamics

The following assumptions were made before a mathematical model was derived from the quadcopter's dynamics (Xuan-Mung & Hong, 2019):

- ✓ The quadrotor's structure is rigid and symmetrical.
- ✓ The quadrotor's Centre of gravity coincides with the origin of the body fixed frame.
- ✓ Propellers and motors are rigid.
- ✓ Thrust and drag forces are proportional to the square of propeller's speed.

3.6.1. Actuator Dynamics

There are four identical propeller and BLDC motor pairs that can generate lift forces to generate the motion of the quadcopter. Two opposite side motors rotate clockwise, while the other pair turn counter-clockwise as shown in Figure 3.5.

There four independent control inputs and six degree of freedom of the drone can be defined as:

- ✓ U_1: throttle input, which is the total force generated by the four propellers.
- ✓ U_2: roll input, which is the moment difference around x-axis due to left-side & right-side propellers.
- ✓ U_3: pitch input, which is the moment difference around y-axis due to front & rear motors
- ✓ U_4: yaw input, which is the net torque exerted on the system around z-axis by the four motors.

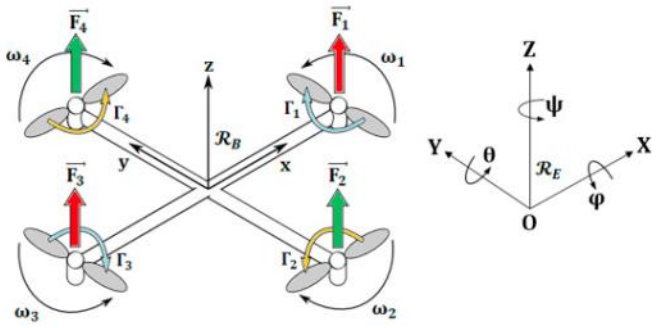

Figure 3.5: The forces & moments generated by the motors.

$$\begin{aligned} U_1 &= f1 + f2 + f3 + f4, \\ U_2 &= f1 + f2 - f3 - f4, \\ U_3 &= f1 - f2 - f3 + f4, \\ U_4 &= f1 - f2 + f3 - f4, \end{aligned} \tag{3.9}$$

Therefore, according to aerodynamic principle, thrust force is proportional to the square of angular velocity of the motor (the square of propeller rotational speed). And the torque is also proportional to the square of the rotational speed of the motor.

$$\begin{cases} F = k\omega^2 \\ \tau = b\omega^2 \end{cases} \tag{3.10}$$

Where k & b are thrust constant of the motors and torque coefficient of the propeller respectively.

3.6.2. Translational equation of motion

The translation equations of motion for the quadrotor are based on Newton's second law and they are derived in the Earth inertial frame (Valavanis & Vachtsevanos, 2015).

$$\begin{aligned} F_N &= R_{IB} f_B - mg - f_D \\ ma = m\ddot{x} &= R_{IB} f_B - \begin{vmatrix} 0 \\ 0 \\ mg \end{vmatrix}, \\ f_B &= \begin{bmatrix} 0 \\ 0 \\ k(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2) \end{bmatrix} = \begin{vmatrix} 0 \\ 0 \\ u_1 \end{vmatrix} \end{aligned} \tag{3.11}$$

$$F_D = k_t \dot{x} = k_t \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \tag{3.12}$$

Solving equations 3.11 and 3.12, the accelerations are obtained as seen in Equation 3.13

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} \frac{u_1}{m}(c(\phi)s(\theta)c(\psi) + s(\phi)s(\psi)) - \frac{k_t}{m}\dot{x} \\ \frac{u_1}{m}(c(\phi)s(\theta)s(\psi) - s(\phi)c(\psi)) - \frac{k_t}{m}\dot{y} \\ \frac{u_1}{m}(c(\phi)c(\theta)) - g - \frac{k_t}{m}\dot{z} \end{bmatrix} \tag{3.13}$$

3.6.3. Rotational Equation of Motion

The rotational motion of the quadcopter from figure 3.3 is described by Euler's equation of rotation (Abdelhay & Zakriti, 2019) in the equation below.

$$\begin{aligned} & I\dot{\omega} + \omega \times (I\omega) = M_b = \tau_B - \tau_G \\ & \tau_B = \begin{bmatrix} u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} -lk & lk & lk & -lk \\ -lk & -lk & lk & lk \\ -b & b & -b & b \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} \\ & \tau_G = \omega \times \begin{bmatrix} 0 \\ 0 \\ J_r\omega_r \end{bmatrix}, ---> \omega_r = \omega_1 - \omega_2 + \omega_3 - \omega_4 \\ & \dot{\omega} = I^{-1}(\tau_B - \tau_G - \omega \times I\omega) \end{aligned} \tag{3.14}$$

where,

I - inertia matrix of quadcopter, τ_B - moment in body frame

J_r - inertia of the rotor τ_G - gyroscopic moment

ω - angular velocity vector in body-fixed frame

M_b - is the moment vector in body frame, and can be derived from the torque vectors. The total torque about the z-axis will be an alternating sum of torques for each propeller.

The symmetrical structure of the quadcopter reduces the inertia matrix, *I*, to a diagonal matrix where the diagonal elements correspond to the mass moment of inertia for each cartesian axis; the off-diagonal elements are products of inertia. Therefore, the inverse of diagonal matrix is the reciprocal of its elements

$$I=\begin{vmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{vmatrix}, I^{-1}=\begin{vmatrix} 1/I_{xx} & 0 & 0 \\ 0 & 1/I_{yy} & 0 \\ 0 & 0 & 1/I_{zz} \end{vmatrix}=\begin{bmatrix} \frac{1}{I_{xx}} \\ \frac{1}{I_{yy}} \\ \frac{1}{I_{zz}} \end{bmatrix} \omega=\begin{vmatrix} p \\ q \\ r \end{vmatrix}, and, I\omega=\begin{vmatrix} I_{xx}p \\ I_{yy}q \\ I_{zz}r \end{vmatrix} \tag{3.15}$$

$$\omega\times I\omega=\begin{bmatrix} i & j & k \\ p & q & r \\ I_{xx}p & I_{xx}q & I_{xx}r \end{bmatrix}=\begin{bmatrix} (I_{zz}-I_{yy})qr \\ (I_{xx}-I_{zz})pr \\ (I_{yy}-I_{xx})pq \end{bmatrix}$$

$$\dot{\omega}=\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix}=\begin{bmatrix} \frac{u_2}{I_{xx}} \\ \frac{u_3}{I_{yy}} \\ \frac{u_4}{I_{zz}} \end{bmatrix}+\begin{bmatrix} \frac{1}{I_{xx}}(I_{yy}-I_{zz})qr \\ \frac{1}{I_{yy}}(I_{zz}-I_{xx})pr \\ \frac{1}{I_{zz}}(I_{xx}-I_{yy})pq \end{bmatrix}+\begin{bmatrix} -\frac{J_r}{I_{xx}}q\omega_r \\ \frac{J_r}{I_{yy}}p\omega_r \\ 0 \end{bmatrix} \tag{3.16}$$

3.6.4. State Space model

The state space model is a mathematical model of a system as a set of input, output, and state variables associated with the equation from the first order deferential equation. Nowadays, it is well known that one of main advantages of the state space method is modelling of multiple-input and multiple-output control system. The state space model is expressed as follows:

$$\begin{cases} \dot{x}=Ax(t)+Bu(t) \\ y=Cx(t)+Du(t) \end{cases} \tag{3.17}$$

Where:

- ✓ x(t) is the state vector of the system,
- ✓ y(t) is output vector,

- ✓ A system matrix,
- ✓ B input matrix,
- ✓ C output matrix and
- ✓ D feedforward matrix.

The first step in representing a system is to select the state vector (contains the state variables); the choice of state variables for a given system is not exceptional. The necessity in choosing the state variables is that they have to be linearly independent (A set of variables is said to be linearly independent if none of the variables can be written as a linear combination of the others) and that a minimum number of them be chosen. Consider 12 state vectors for 6DOF quadcopter, the states are position, orientation and rate of change of position and orientation.

$x = [x, y, z, \dot{x}, \dot{y}, \dot{z}, \phi, \theta, \psi, p, q, r]^T$.

Let take $a_1 = \frac{I_{yy} - I_{zz}}{I_{xx}}, a_2 = \frac{J_r}{I_{xx}} a_3 = \frac{I_{zz} - I_{xx}}{I_{yy}}, a_4 = \frac{J_r}{I_{yy}}, a_5 = \frac{I_{xx} - I_{yy}}{I_{zz}} = 0$

$b_1 = \frac{1}{I_{xx}}, b_2 = \frac{1}{I_{yy}}, b_3 = \frac{1}{I_{zz}}$

from equation (3.13) and (3.16), we can simplify the equation to

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} \frac{u_1}{m}(c(\phi)s(\theta)c(\psi) + s(\phi)s(\psi)) \\ \frac{u_1}{m}(c(\phi)s(\theta)s(\psi) - s(\phi)c(\psi)) \\ \frac{u_1}{m}(c(\phi)c(\theta)) - g \end{bmatrix} \quad \text{linear acceleration} \quad (3.18)$$

$$\dot{\omega} = \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{u_2}{I_{xx}} \\ \frac{u_3}{I_{yy}} \\ \frac{u_4}{I_{zz}} \end{bmatrix} + \begin{bmatrix} a_1 qr \\ a_3 pr \\ a_5 pq \end{bmatrix} \quad \text{angular acceleration} \quad (3.19)$$

Where $x_1 = x, x_2 = y, x_3 = z, \dot{x}_1 = x_4 = \dot{x}, \dot{x}_2 = x_5 = \dot{y}, \dot{x}_3 = x_6 = \dot{z}$
$x_7 = \phi, x_8 = \theta, x_9 = \psi, x_{10} = p, x_{11} = q, x_{12} = r,$ from the above two equations

$$
\begin{cases}
\dot{x}_4 = \ddot{x} = \dfrac{u_1}{m}(c(\phi)s(\theta)c(\psi) + s(\phi)s(\psi)) \\
\dot{x}_5 = \ddot{y} = \dfrac{u_1}{m}(c(\phi)s(\theta)s(\psi) - s(\phi)c(\psi)) \\
\dot{x}_6 = \ddot{z} = \dfrac{u_1}{m}(c(\phi)c(\theta)) - g, \\
\dot{x}_7 = p + s(\phi)t(\theta)q + c(\phi)t(\theta)r \\
\dot{x}_8 = qc(\phi) - rs(\phi) \\
\dot{x}_9 = q\dfrac{s(\phi)}{c(\theta)} + r\dfrac{c(\phi)}{c(\theta)} \\
\dot{x}_{10} = \dot{p} = a_1 qr + b_1 u_2, \\
\dot{x}_{11} = \dot{q} = a_3 pr + b_2 u_3, \\
\dot{x}_{12} = \dot{r} = a_5 pq + b_3 u_4,
\end{cases}
\tag{3.20}
$$

The full nonlinear model is very useful, as it provides insight into the behavior of the vehicle. However, a linear model is used widely, and needs low computational power. Typically, the linearization of a nonlinear state space model is executed at an equilibrium point of the model using the Taylor Series expansion (Zhang et al., 2014). Typically, the linearization of a nonlinear state space model $x = f(x) + g(x)U$ is executed at an equilibrium point of the model

$$
\begin{aligned}
& f(x^*, U^*) = 0, \\
& A = \left.\frac{\delta f}{\delta x}\right|_{x=x^*},
\end{aligned}
\tag{3.21}
$$

As the hovering is one of the most important regimes forming a quadrotor, at this point, the condition of equilibrium of the quadrotor in terms of equation (3.21) is given as in

$$
\begin{aligned}
& u_1^* = mg, \\
& u_2^* = u_3^* = u_4^* = 0, \\
& x_1^* = x_2^* = x_6^* = x_{12}^* = C,
\end{aligned}
\tag{3.22}
$$

where C is a random constant all other states are taken as zero at the hovering state. While hovering, some assumptions are reasonable, such as small velocities and rotational velocities. To

find equilibrium points for $[x, y, z, \phi, \theta, \psi]$, By performing a Taylor series expansion and eliminating the higher-order terms on equation (3.20), and using small-angle approximations, a linear model is given:

$$\begin{cases} \ddot{x} = g\theta \\ \ddot{y} = -g\phi \\ \ddot{z} = \dfrac{u_1}{m} - g \\ \ddot{\phi} = \dfrac{u_2}{I_{xx}} \\ \ddot{\theta} = \dfrac{u_3}{I_{yy}} \\ \ddot{\psi} = \dfrac{u_4}{I_{zz}} \end{cases} , where, \begin{cases} \phi = \theta = \psi \approx 0 \\ \sin(\theta) \approx \theta \\ \cos(\theta) \approx 1 \\ \tan(\theta) \approx \theta \\ \sin^2(\theta) \approx 0 \end{cases} i, e \begin{cases} \ddot{\phi} = \dot{p}, \ddot{\theta} = \dot{q}, \ddot{\psi} = \dot{r}, \end{cases} \tag{3.23}$$

Finally, the linearized state space representation around equilibrium points become:

$$\begin{cases} \dot{x} = Ax(t) + Bu(t) \\ y = Cx(t) + Du(t) \end{cases}$$

$$A = \begin{vmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & g & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -g & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{vmatrix}, B = \begin{vmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{m} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & \frac{1}{I_{xx}} & 0 & 0 \\ 0 & 0 & \frac{1}{I_{yy}} & 0 \\ 0 & 0 & 0 & \frac{1}{I_{zz}} \end{vmatrix} \tag{3.24}$$

$$C = \begin{vmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{vmatrix}, D = \begin{vmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{vmatrix} \tag{3.25}$$

3.7. Controllability and observability

To perform various tasks with a Quadcopter, first we have to check its controllability, observability as well as its stability.

3.7.1. Controllability

A system is controllable if there exists a control input u(t) that transfers any state of the system to desire state in finite time (Kalman et al., 1960). An LTI system is aid to be controllable if and only if its controllability matrix - CO, has full rank, therefore the quadcopter model used in this paper is controllable by testing the controllability on MATLAB.

that is rank (CO) = n, where n is the number of states:

$$CO = \left| B \quad AB \quad A^2B \quad --- \quad A^{n-1}B \right|$$

3.7.2. Observability

When some state variables are unmeasurable, we need to check, if they can be estimated from the model (observer design). An LTI system is said to be observable if and only if its observability matrix - OB, has full rank,

that is rank (OB) = n, where n is the number of states

CHAPTER FOUR

4. ESTIMATOR AND CONTROLLER DESIGN

4.1. Chapter Overview

The basic functionality of unmanned aerial vehicle is to is to navigate from its current location to a designated goal. To perform this task autonomously requires a navigation system that is composed of the standard elements of state estimation, control, mapping and planning. Each of these blocks builds on top of the previous ones in order to construct the full navigation system. For example, the controller requires a working state estimator while the planner requires a working state estimator, controller and mapping system. In this chapter the principle & detailed description of Model Predictive Control (MPC), and Kalman filter (KF) is covered, followed by their design.

4.2 Estate Estimation

In order to perform any navigation task, the quadcopter needs to know information on the current state. Thus, the state estimation task for our drone consists of estimating the position and orientation, of the drone with respect to an inertial reference frame. A state estimator is used to estimate the essential states of the quadcopter from the sensor measurements.

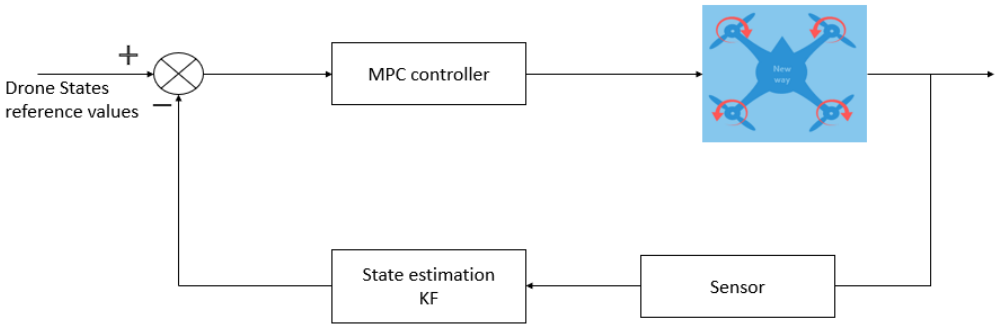

Figure 4.1: State estimation and controller

4.2.1. Kalman Filter (KF)

A Kalman filter is an iterative mathematical process that uses a set of equations and consecutive data inputs to quickly estimate the true value of the states of a system being measured; which contain random error or uncertainty (Jiang et al., 2014). Kalman filters are very popular due to

their ease of use and good performance. Even though the Kalman filter was originally proposed only for linear systems , Extended Kalman filter (EKF) extends this estimation ability to non-linear systems (Abeywardena et al., 2015).

The goal of the Kalman filter is to

- ✓ Estimate the state variable that cannot be directly measured and
- ✓ Filter out the measurement noise

Assuming that the process model is described by a linear time-varying (LTV) model in discrete time, such that

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + v(k) \\ y(k) &= Cx(k) + Du(k) + w(k) \end{aligned} \tag{4.1}$$

Where

- ✓ x(k) is the state,
- ✓ u(k) is the input, y(k) is the output,
- ✓ w(k) is the process noise, and v(k) is the measurement noise. With zero mean and known covariance matrices Q(k), and R(k) respectively.

The problem is as follows: Given a measurement arrangement and an input order up to time k, which delivers the finest estimate of the state vector x such that $x(k+m), given\{y(i), u(i) \mid i \le k\}$

- ➢ If we have m < 0, we call it a smoothing problem,
- ➢ if m=0 we call it a filtering problem, and
- ➢ if m > 0 we call it a prediction problem

We will here only discuss filtering and prediction, which are the most common applications in control engineering. To solve the estimation problem, a model of the noise v(k) and w(k) are needed.

The Kalman filter has an iterative structure. It has two types of states: Prediction based on last estimate: and Update prediction based on the state given the measurement. There is a corrector step where the most current measurement is well thought-out, and there is a prediction stage for the next time instant.

1. Prediction based on last estimate:

$$\begin{aligned}\hat{x}(k+1|k) &= A(k)\hat{x}(k|k) + B(k)u(k)\\ P(k+1|k) &= AP(k|k)A^T + Q\\ \hat{y}(k) &= C(k)\hat{x}(k+1|k)\end{aligned} \tag{4.2}$$

2. Calculate correction based on prediction and current measurement:

$$\Delta x = f\left(y(k+1), \hat{x}(k+1|k)\right) \tag{4.3}$$

3. Update prediction:

$$\hat{x}(k+1|k+1) = \hat{x}(k+1|k) + \Delta x \tag{4.4}$$

Calculating the correction: $\Delta x = C^T\left(CC^T\right)^{-1}\left(y(k+1) - C\hat{x}(k+1|k)\right)$

1. **Prediction** Let start from x(k\k) and p(k\k)

$$\begin{aligned}&\hat{x}(k+1|k) = A\hat{x}(k|k) + Bu(k)\\ &P(k|k) = E\left((x(k) - \hat{x}(k|k))(x(k) - \hat{x}(k|k))^T\right) \text{ and}\\ &P(k+1|k) = E\left((x(k+1) - \hat{x}(k+1|k))(x(k+1) - \hat{x}(k+1|k))^T\right)\end{aligned} \tag{4.5}$$

To make computation easier, lets shift notation

$$\begin{aligned}P_{k+1}^- &= E\left((x_{k+1} - \hat{x}_{k+1}^-)(x_{k+1} - \hat{x}_{k+1}^-)^T\right)\\ &= E\left(\left(A\left(x_k - \hat{x}_k\right) + v_k\right)\left(A\left(x_k - \hat{x}_k\right) + v_k\right)^T\right)\\ &= AE\left(\left(x_k - \hat{x}_k\right)\left(x_k - \hat{x}_k\right)^T\right)A^T + E\left(v_k v_k^T\right)\\ &= AP_kA^T + Q\\ P(k+1|k) &= AP(k|k)A^T + Q\end{aligned} \tag{4.6}$$

2. Computing the correction

From step 1 we get $\hat{x}(k+1|k)$ and $P(k+1|k)$. Now we use these to compute Δx:

$$\begin{aligned}&\Delta x = P(k+1|k)C\left(CP(k+1|k)C^T\right)^{-1}\left(y(k+1) - C\hat{x}(k+1|k)\right)\\ &let, \Delta x = W\nu\\ &W = P(k+1|k)H\left(CP(k+1|k)C^T\right)^{-1}\\ &\Delta x = W\left(y(k+1) - C\hat{x}(k+1|k)\right)\end{aligned} \tag{4.7}$$

3. Update

$$\hat{x}(k+1|k+1) = \hat{x}(k+1|k) + \Delta x$$
$$\hat{x}(k+1|k+1) = \hat{x}(k+1|k) + W\nu$$
$$\hat{\mathbf{x}}_k^- = A\hat{\mathbf{x}}_{k-1} + B\mathbf{u}_k \quad (4.8)$$
$$P(k+1|k+1) = P(k+1|k) - WCP(k+1|k)C^T W^T$$

$$P_k^- = AP_{k-1}A^T + Q$$
$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + K_k(z_k - C\hat{\mathbf{x}}_k^-)$$
$$where, z_k - C\hat{\mathbf{x}}_k^-, \text{innovation}$$
update error covariance matrix
$$P_k = (I - K_k C) P_k^- \quad (4.9)$$
The optimal Kalman gain K_k is
$$K_k = P_k^- C^T (CP_k^- C^T + R)^{-1}$$
$$K_k = \frac{P_k^- C^T}{CP_k^- C^T + R}$$

4.2. Model Predictive Control

Model predictive control (MPC) is an optimal model-based control method to optimize control inputs by minimizing an objective function. The objective function is well-defined based on both present and predicted system variables and is assessed using clear model to predict future system outputs. This section explains the present practice in MPC, computational and design issues. Model predictive control integrates ideas from systems theory, system identification and optimization. Model Predictive Control (MPC), also known as Moving Horizon Control (MHC) or Receding Horizon Control (RHC), is a prevalent method for the control of dynamical systems

In model predictive control, a mathematical model of the dynamic system to be controlled is utilized to forecast the model output, $Ym(k + i)$, of the system over a horizon known as the **prediction horizon**, *ny* (Gurbani et al., 2017). The chosen sample time is *i* for each prediction step, where the number of steps is corresponding to the size of the prediction horizon. An arrangement of control inputs is obtained by minimizing a cost function of the error among the predicted outputs and the reference or set point values $r(k + i)$, and a weighted control input term $u(k + i)$ over a **control horizon**, n_u (Robinson & Cima, 2017) The quantity of control inputs in

this sequence is equal to the size of the control horizon. These arrangement of control inputs are the control actions required to drive the model and system to the reference values. Only the first element of the control input sequence is implemented, both in the model and the process, as the system measurement and reference values are continuously being updated in succeeding sampling instants.

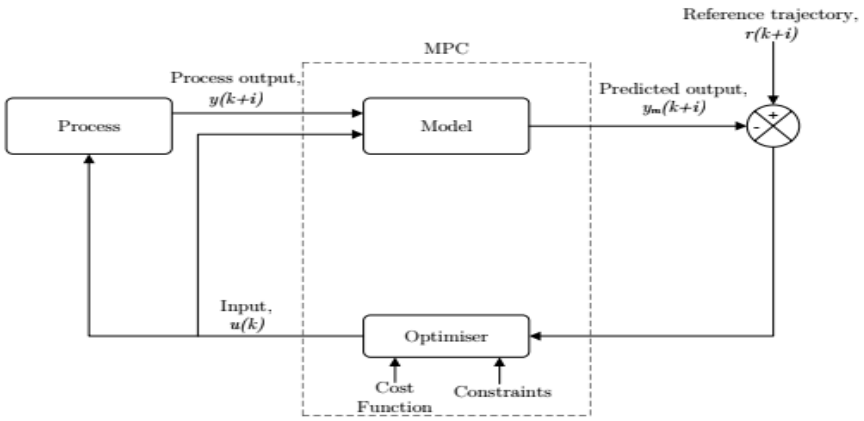

Figure 4.2: Steps of MPC

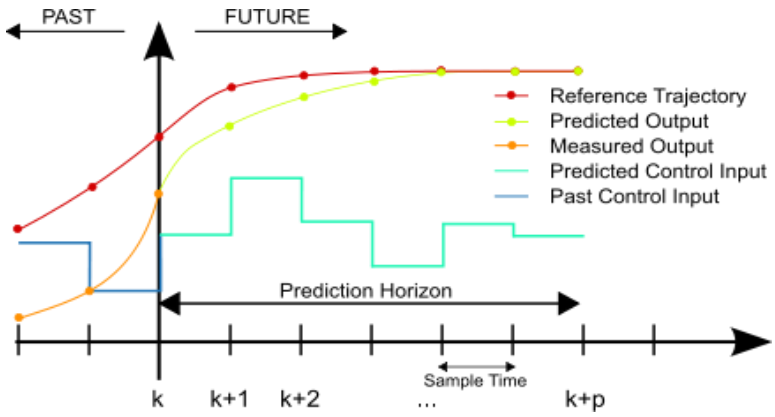

Figure 4.3: Prediction Horizon

4.2.1. Output Prediction

Using the quadcopter model $x(k+1) = f(x(k), u(k))$, the upcoming state and output predictions are considered recursively in this manner, the model predictive controller calculate the predicted plant output with the future control signal as the adjustable variable(s). This prediction is defined within a preselected prediction horizon (Wang, 2009). Taking *k* as the present sample time, the future control route is denoted by,

$$\Delta U = \left[\Delta u(k);\ \Delta u(k\ +\ 1);\ \Delta u(k+\ 2);\ \cdots;\ \Delta u(k\ +\ n_u\text{-}1)\right]^T \tag{4.10}$$

The control horizon is preferred to be less than or equal to prediction horizon. The forthcoming state variables are essential in order to compute the predicted plant output. The future state variables are represented as: $x(k + 1)$; $x(k + 2)$; $x(k + 3)$; $\cdots$; $x(k + ny)$

$$\begin{aligned}
&at,(k+1),\\
&x(k+1) = Ax(k) + B\Delta u(k)\\
&y(k+1) = Cx(k+1)\\
&at,(k+2),\\
&x(k+2) = Ax(k+1) + B\Delta u(k+1)\\
&y(k+2) = Cx(k+2)\\
&x(k+2) = A(Ax(k) + B\Delta u(k)) + B\Delta u(k+1)\\
&x(k+2) = A^2x(k) + AB\Delta u(k) + B\Delta u(k+1)\\
&y(k+2) = C(A^2x(k) + AB\Delta u(k) + B\Delta u(k+1))
\end{aligned} \tag{4.10}$$

This procedure continues recursively to give n_y-step forward predictions. Hence, the output for n_y-step forward predictions can be expressed as,

$$y(k+n_y) = C\left[(A^{n_y}x(k) + A^{n_y-1}B\Delta u(k) + A^{n_y-2}B\Delta u(k+2) + ... + B\Delta u(k+n_y-1)\right] \tag{4.11}$$

The future output predictions are denoted as,

$$Y = \left[y(k+1), y(k+2), y(k+3), ..., y(k+n_y)\right]^T \tag{4.12}$$

$$\begin{bmatrix} y(k+1)\\ y(k+2)\\ y(k+3)\\ .\\ .\\ .\\ y(k+n_y)\end{bmatrix} = \begin{bmatrix} CA\\ CA^2\\ CA^2\\ .\\ .\\ .\\ CA^{n_y}\end{bmatrix} x(k) + \begin{bmatrix} \mathrm{CB} & 0 & 0 & ...0\\ \mathrm{CAB} & \mathrm{CB} & 0 & ...0\\ \mathrm{CA^2B} & \mathrm{CAB} & \mathrm{CB} & ...0\\ . & . & . & .\\ . & . & . & .\\ . & . & . & .\\ \mathrm{CA^{n_y-1}B} & \mathrm{CA^{n_y-2}B} & \mathrm{CA^{n_y-3}B} & ....\mathrm{CA^{n_y-n_u}B}\end{bmatrix} \Delta U \tag{4.13}$$

Simply,

$$Y = Px(k) + H\Delta U$$

where,

$$P = \begin{bmatrix} CA \\ CA^2 \\ CA^2 \\ . \\ . \\ . \\ CA^N \end{bmatrix},$$

$$H = \begin{bmatrix} \mathrm{CB} & 0 & 0 & ...0 \\ \mathrm{CAB} & \mathrm{CB} & 0 & ...0 \\ \mathrm{CA^2B} & \mathrm{CAB} & \mathrm{CB} & ...0 \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ \mathrm{CA^{N-1}B} & \mathrm{CA^{N-2}B} & \mathrm{CA^{N-3}B} & ....\mathrm{CA^{N-M}B} \end{bmatrix} \quad (4.14)$$

4.2.2. Constraints

MPC is considered in part by its ability to integrate constraints. In practice, all systems are subject to operative constraints such as limited dimensions and restricted control capacity. In numerous situations, these constraints are intentionally performed and are proposed to be made as tight as possible in order to decrease energy utilization, to minimize the operation of resources or to simply decrease the size of a certain device (Gao, 2014) Constraints can be performed on the control variable (also known as the input), *u(k)*, rate of input, $\Delta u(k)$ and also the output, *y(k)* or state variables, *x(k)*. Considering the case of the quadcopter, which attains control by changing the angular speeds of each motor, the speeds differ from a smallest value of zero to a extreme value that is reliant on the motor specification. In the practical case u_i is the voltage and y_i is the state (position ad velocity)

$$\begin{aligned} y_{\min} \le y_i \le y_{\max} \\ x_{\min} \le x_i \le x_{\max} \\ u_{\min} \le u_i \le u_{\max} \\ \Delta u_{\min} \le \Delta u_i \le \Delta u_{\max} \end{aligned} \quad (4.15)$$

As there are four control variables needed to control the translation and rotational motion of the quadcopter, each control variable is subjected to constraints as shown below,

$$
\begin{aligned}
u_1^{\min} &\le u_1 \le u_1^{\max} \\
u_2^{\min} &\le u_2 \le u_2^{\max} \\
u_3^{\min} &\le u_3 \le u_3^{\max} \\
u_4^{\min} &\le u_4 \le u_4^{\max}
\end{aligned}
\tag{4.16}
$$

The calculation of the constraints is based on the quadrotor extreme speeds of the motors. The quadcopter used in this thesis is assumed with four brushless 2200Kv motors. The maximum speed evaluation of the motors was attained from the manufacturer datasheet by matching the propeller category, propeller dimensions, electrical energy and current of the battery used in the quadcopter with the data accessible on the website. Therefore, from these datasheets we gate the maximum speed rating to be 7720 rpm, thus the constraints executed in the MPC design are constraints on the input and the rate of input change. In order to limit these constraints, the total force generated by the motors required to be used. The mass of the quadcopter (battery included) was read from a mass scale to be 1.5kg. using the acceleration due to gravity as 9.81 m/s^2, then calculating the weight of the quadcopter will be easy.

$$\text{Weight} = 1.5 \times 9.81 = 14.72 \text{ N}$$

And thrust of the quadrotor calculated as follows,

$$f = \sum_{i=1}^{4} k\omega_i^2$$

From table 5.2, specification of the motor the thrust constant is 2.984×10^{-5}N/rpm^2. Then the speed for individual quadrotor motor was calculated as follows,

$$
\begin{aligned}
14.72 &= k\sum_{i=1}^{4} \omega_i^2 \\
\omega_i &= \sqrt{\frac{14.72}{2.984 \times 10^{-5}}} \\
&\approx 6013\ rpm\ , \text{minimum speed}
\end{aligned}
\tag{4.17}
$$

As formulated in chapter 3 of the mathematical modeling, the input for the rotational motion of the quadrotor are torques in the roll, pitch and yaw axes of the quadrotor. These torque equations are computed below,

$$\tau_B = \begin{bmatrix} u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} -lk & lk & lk & -lk \\ -lk & -lk & lk & lk \\ -b & b & -b & b \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} \tag{4.17}$$

where U_2, U_3 and U_4 are torques $\tau\varphi$, $\tau\theta$ and τ respectively. The input constraints are the extreme torques and are formulated below. For instance, in formulating the maximum rolling torque, $\tau\varphi max$, the maximum motor speed was replaced into positive motors and the lowest thrust motor speed replaced into negative motors. Therefore, the extremes of motor speeds are 7720 rpm and 6013 rpm correspondingly. The supreme torques are calculated below,

$$\begin{aligned} &\tau_\phi max = 0.243 \times 4.0687 \times 10^{-7}(2*(7720^2 - 6013^2)) \\ &\tau_\phi max = 0.243 \times 4.0687 \times 10^{-7}(46884462) \\ &\tau_\phi max = 4.63 \\ &therefore, \mathrm{U}^{\max} = \left[4.63,\ 4.63, 0.245\right]^T \\ &and, \mathrm{U}^{\min} = \left[-4.63,\ -4.63, -0.245\right]^T \end{aligned} \tag{4.18}$$

4.2.3. Optimization

The design of a predictive control scheme is to analyze the predicted plant output with the upcoming control signal as the modifiable variables. This prediction is labeled within an optimization window. The model predictive control is used to formulate the predicted outputs to the wanted reference signals. This problem is realized by minimizing a cost function that is contained the error between predicted state and reference signal, based on constraints on the process. The output from this optimization is a sequence of inputs, ΔU, used to calculate the predicted outputs to the reference path. At the sample time, k, output predictions are produced consistent with the size of the prediction horizon (Collares Pereira, 2020).

An optimization problem consists of three components.

- ✓ An objective function, $\phi(\mathbf{w})$, that shall be minimized or maximized,

- ✓ decision variables, **w**, that can be chosen, and
- ✓ constraints that shall be respected, e.g. $\boldsymbol{g}_1(\boldsymbol{w}) = 0$ (equality constraints) or $\boldsymbol{g}_2(\boldsymbol{w}) \geq 0$ (inequality constraints).

The mathematical formulation of optimization problem in Standard Form is **Nonlinear Programming Problem** (**NLP**) which is A standard problem formulation in numerical optimization.

$$\min_{w} \Phi(w) \qquad \text{Objective function}$$

$$\text{s.t. } g_1(\text{w}) \leq 0, \text{ and } g_2(\text{w})=0 \qquad \text{inequality and equality constraints}$$

Nonlinear Programming Problem (NLP) can be:

- ✓ **Linear Programming (LP)** (when, Φ, g_1 and g_2 are linear, i.e. these functions can be expressed as linear combinations of the elements of **w**).
- ✓ **Quadratic Programming (QP)** (when g_1 and g_2 are affine, but the objective Φ is a linear-quadratic function).

For a given objective function maximization can be treated as a minimization of the negative objective.

Minimization

$$\min_{w} \Phi(w)$$

$$\text{s.t. } g_1(\text{w}) \leq 0, \text{ and } g_2(\text{w})=0$$

Maximization

$$\min_{w} \Phi(w) \equiv \begin{cases} \min_{w} -\Phi(w) \\ \text{s.t. } g_1(\text{w}) \leq 0, \text{ and } g_2(\text{w})=0 \end{cases} \qquad (4.19)$$

$$\text{s.t. } g_1(\text{w}) \leq 0, \text{ and } g_2(\text{w})=0$$

4.2.4. MPC Mathematical Formulation

Running (stage) Costs: characterizes the control objective

$$l(x,u) = \| x_u - x^r \|^2_Q + \| u - u^r \|^2_R \tag{4.20}$$

Cost Function: Evaluation of the running costs along the whole prediction horizon:

$$J_N(x,u) = \sum_{k=0}^{N-1} l(x_u(k), u(k)) \tag{4.21}$$

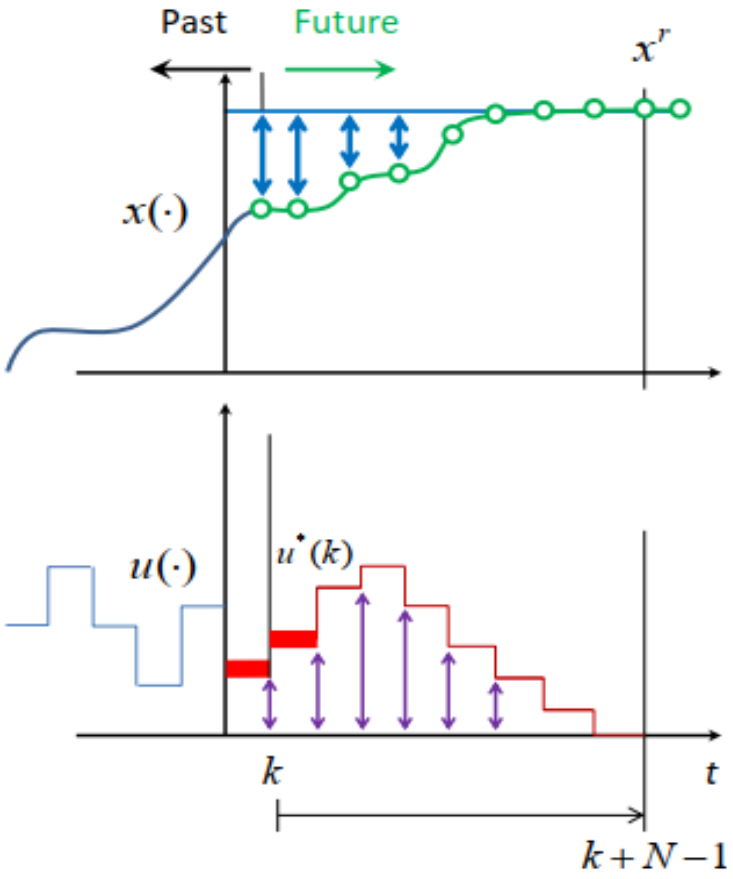

Figure 4.4: Prediction and optimization

Optimal Control Problem (OCP): is formulated to find a minimizing control sequence

$$
\begin{aligned}
&\underset{u}{\text{minimize}}\ J_N(x_0,u)=\sum_{k=0}^{N-1} l(x_u(k),u(k)) \\
&\text{subject to}\ :\ \mathrm{x}_u(\mathrm{k+1})=f(x_u(k),u(k)) \\
&x_u(0)=x_0, \\
&u(k)\in U,\forall k\in[0,N-1] \\
&x_u(k)\in X,\forall k\in[0,N]
\end{aligned}
\tag{4.22}
$$

CHAPTER FIVE

5. RESULT AND ANALYSIS

In this Chapter, the validity and stabilization of the proposed State estimator (Kalman filter) and Model Predictive controller (MPC) for the Quadcopter dynamics are simulated using MATLAB/Simulink environment and physical simulator which is gazebo for software in the loop simulation. The results of the execution of MPC for position and angular rate tracking of a quadcopter UAV is evaluated based on the error deviation criteria. In addition to tracking, we also analyzed its adaptiveness for the disturbance variation of the system. For the simulation, the following parameters shown in Table 5.1 are used from our experimental setup at ASTU space technology institute and my own drone specification:

Table 5. 1: Motor Specification and rating

Specification	
Model	A2212
Motor KV (rpm/v)	EMAX 630kv
Maximum efficiency	80%
LiPo Batteries	2S-3S
Minimum ESC Specification	18A (30A Suggested)
Weight (gm)	48
Thrust @ 3S with 1045 propeller	1200 gms
Thrust @ 3S with 9045 propeller	750 gms
Thrust @ 3S with 0845 propeller	650 gms

Table 5. 2: Physical parameter for quadcopter

PARAMETERS	***VALUE AND UNIT***
arm length (l)	*0.5m*

total mass (m)	*0.1-6kg*
body inertia of x-axis (I_x)	*0.022*
body inertia of y-axis (I_y)	*0.022*
body inertia of z-axis (I_z)	*0.028*
motor inertia (J_r)	*2.8385×10^{-5} N.m/rad/s2*
thrust coefficient (k)	*2.9842×10^{-5} N.m/rad/s*
drag coefficient (d)	*3.2320×10^{-7} N.m/rad/s*
gravitational (g)	*9.8m/s^2*

5.1. Proposed Control System Result

The MPC controller designed in the preceding chapter was simulated in MATLAB in order to test its performance. A sinusoidal reference signal was used as the orientation to be tracked by the controller. Its performance was observed by plotting the model response together with the references, by changing the control and prediction horizons. The weights on the control inputs were adjusted to fine-tune these variations. Additionally, the appropriate matrices and vectors, based on the size of the control and prediction horizons, were altered for well-matched matrix and vector computations. The sinusoidal signal tracked by the MPC angular rates controller was formulated below,

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} \sin(t)+\cos(3t)/2 \\ \sin(t)+\cos(3t)/2 \\ \sin(t)+\cos(2t)/2+\cos(3t)/3 \end{bmatrix}$$

The variables $\dot{\phi}$, $\dot{\theta}$ and $\dot{\psi}$ are the angular rates for the roll, pitch and yaw axes respectively measured in rad/s. t is the simulation time step which is restricted between 0 and 20; with raises of 0.2. The table below shows that the initializations of the parameters required for the MATLAB simulations, where $[p,q,r]=[\dot{\phi},\dot{\theta},\dot{\psi}]^T$ The first simulation result is neglecting disturbance factor

Table 5. 3: MATLAB MPC simulation parameters for $n_u = 3$, $n_y = 6$

n_u	ny		p & q	r	Disturbance
3	6		$\sin(t)+\cos(3t)/2$	$\sin(t)+\cos(2t)/2+\cos(3t)/3$	0

5.2. Trajectory Tracking Performance without Disturbance

5.2.1. Roll rate controller tracking performance

In Figure 5.1, the angular rates for the roll axis trajectories to the desired trajectory starting from the initial up to the final.

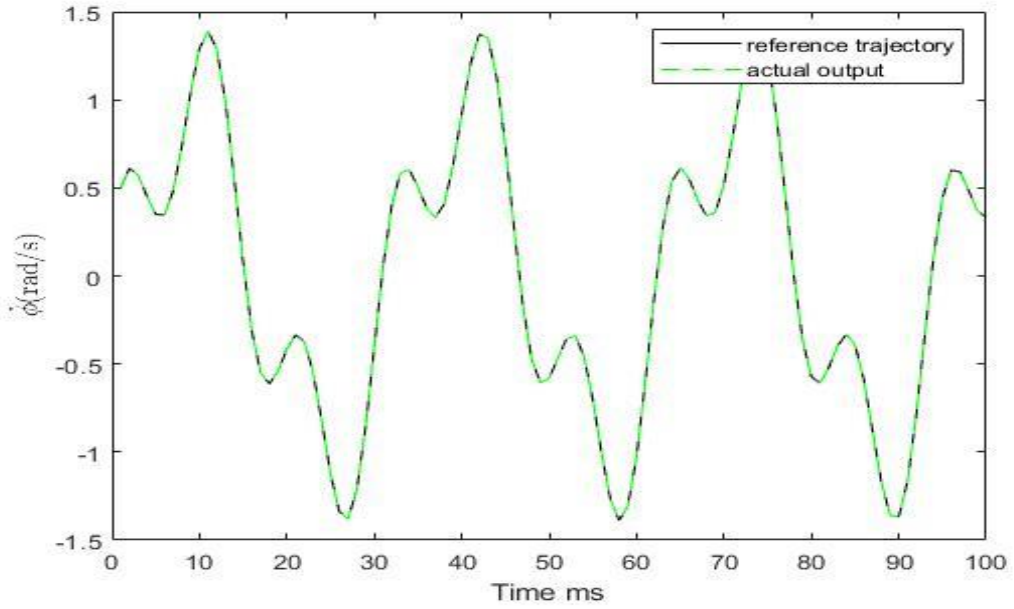

Figure 5. 1: roll angle angular rate trajectory

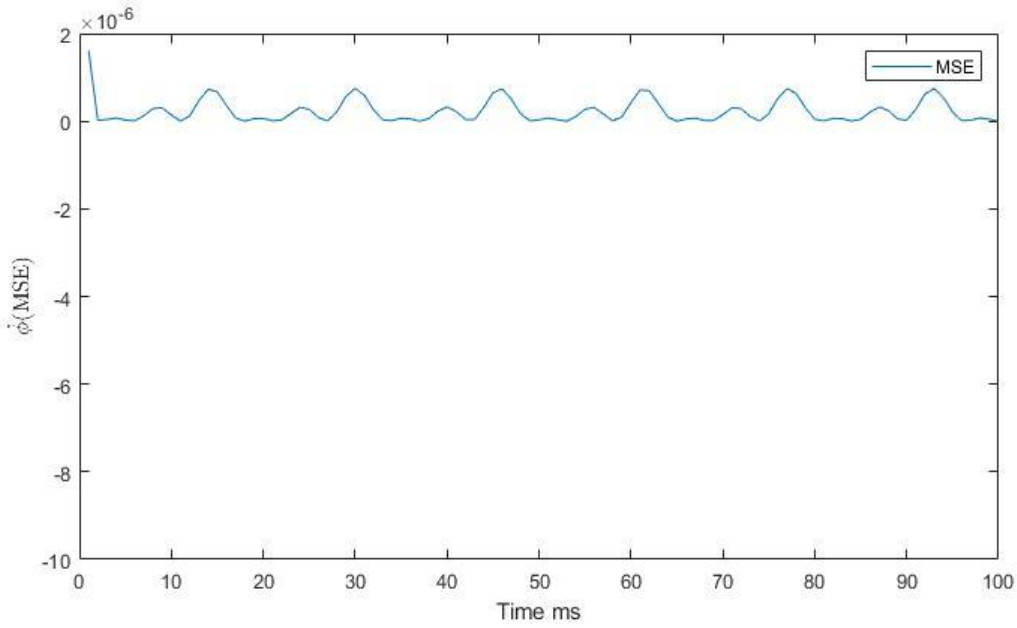

Figure 5. 2: Roll rate controller tracking error

5.2.2. Pitch rate controller tracking performance

As shown in Figure 5.3, the actual trajectories of pitch rate track the given desired trajectories starting from the initial up to the final within the given simulation time. As X position and pitch angle is related internally the small pitch reference trajectory is generated. Therefore, the actual pitch rate follows the generated trajectory.

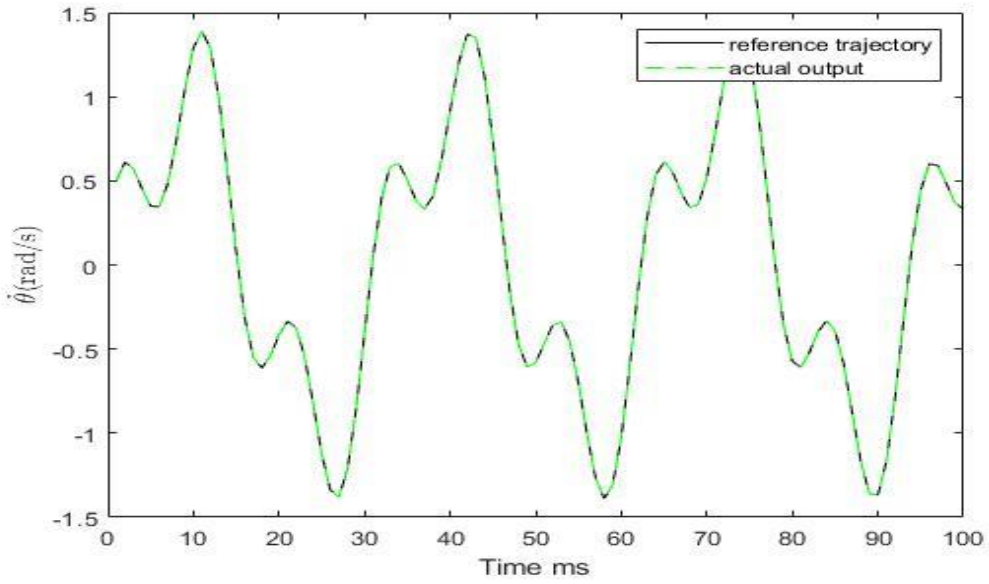

Figure 5. 3: Pitch rate reference trajectory tracking controller using MPC

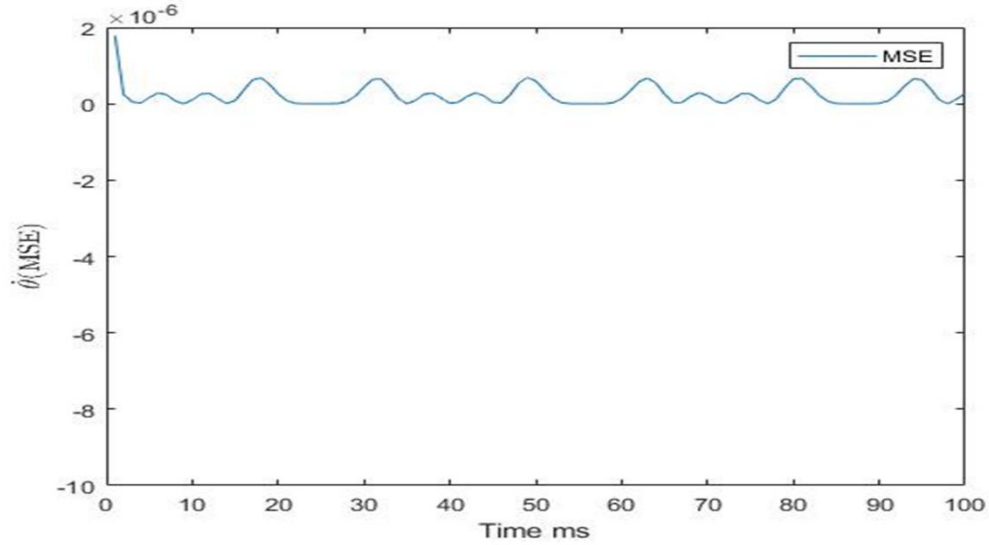

Figure 5. 4: pitch rate controller tracking error

5.2.3. Yaw rate controller tracking performance

In Figure 5.5 the yaw rate tracking is desired trajectories starting from the initial up to the final within the given simulation time.

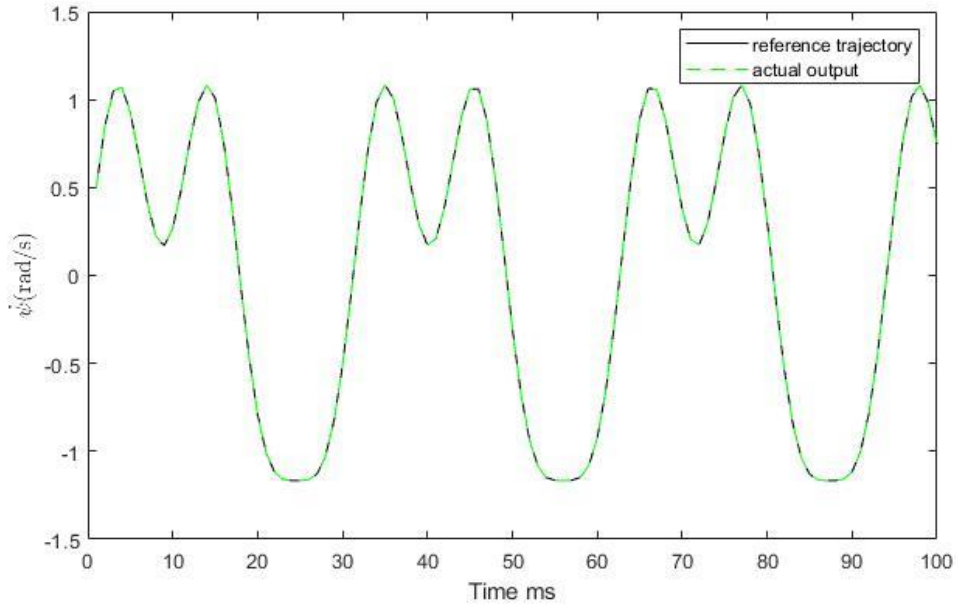

Figure 5. 5:Yaw rate reference trajectory controller performance using MPC

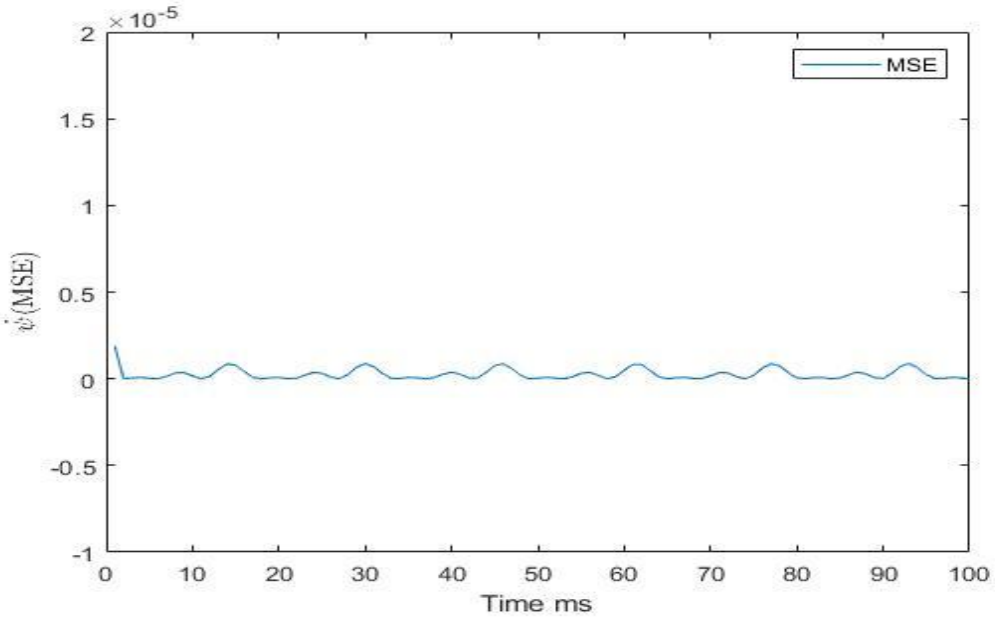

Figure 5. 6: Yaw rate controller tracking error

5.3. Trajectory Tracking Performance with Disturbance

Table 5. 4: MATLAB MPC simulation parameters for $n_u = 3$, $n_y = 6$, with disturbance

n_u	ny	p & q	r	Disturbance

3	6	$\sin(t)+\cos(3t)/2$	$\sin(t)+\cos(2t)/2+\cos(3t)/3$	rand(3,101)*2 -1)*0.5

5.3.1. Roll rate controller tracking performance

In Figure 5.7 the roll rate tracking and from figure 5.8 we can clearly see that mean square error is 0.0071 which shows that even with disturbance the quadcopter traces the reference roll rate efficiently.

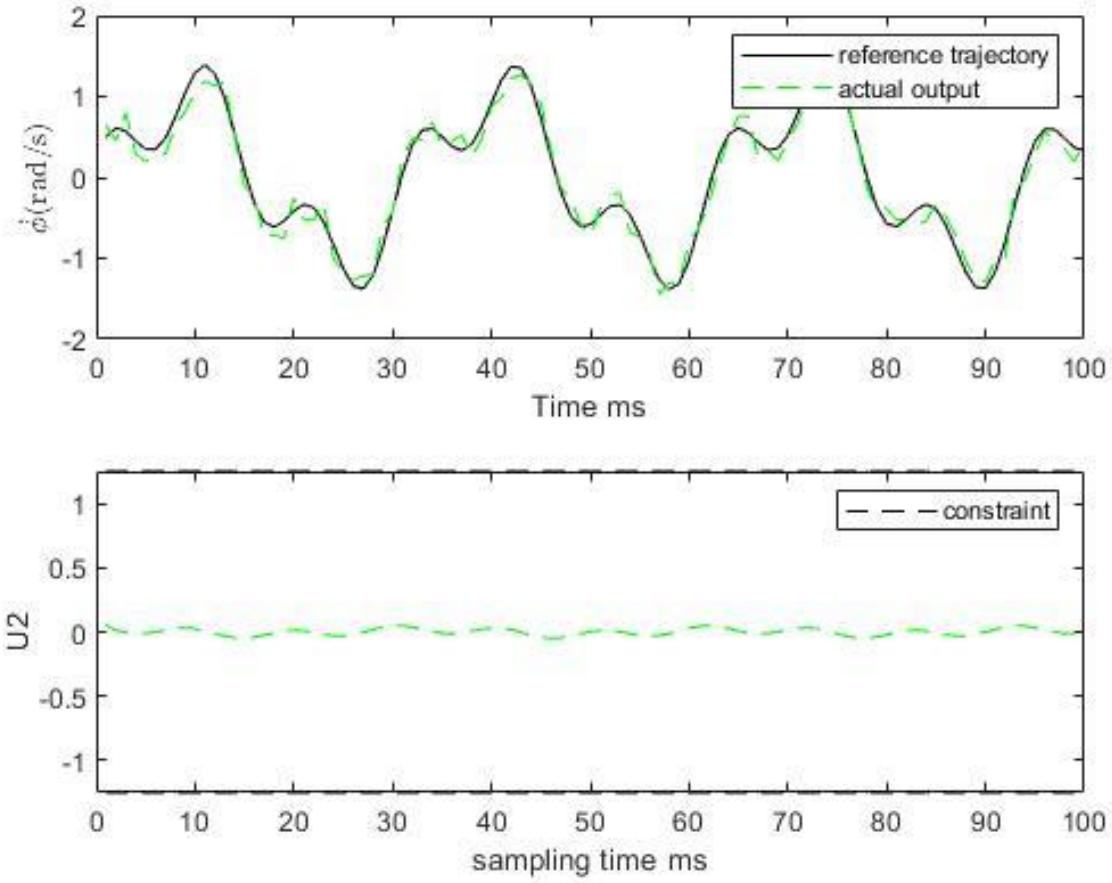

Figure 5. 7: MATLAB roll rates

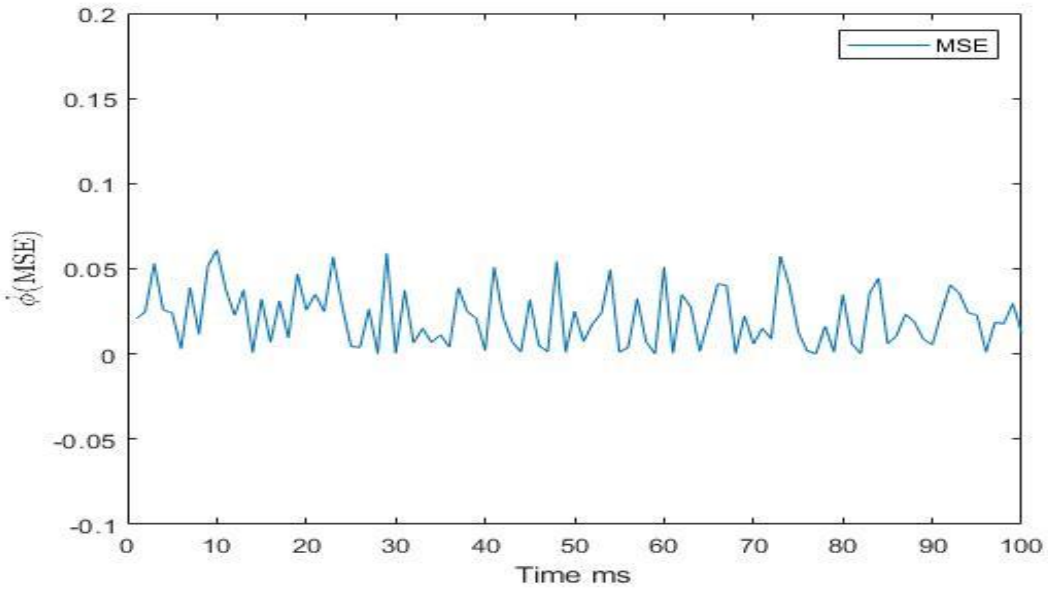

Figure 5. 8: Roll rate controller tracking error

5.3.2. Pitch rate controller tracking performance

In Figure 5.9 the pitch rate tracking and from figure 5.10 we can clearly see that mean square error is 0.0071 which is equal with that of roll rate which shows that even with disturbance the quadcopter traces the reference roll rate efficiently.

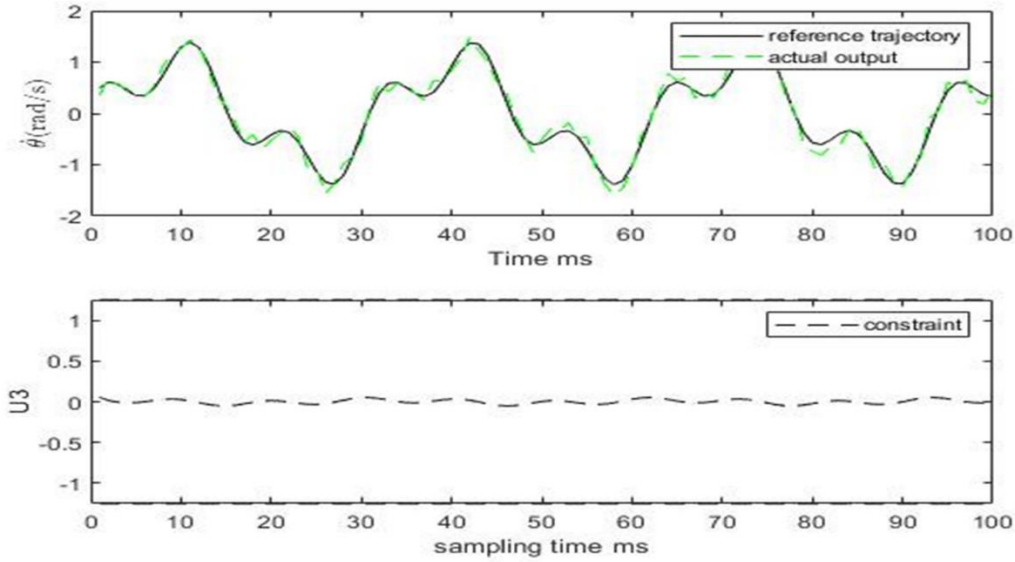

Figure 5. 9: MATLAB pitch rates

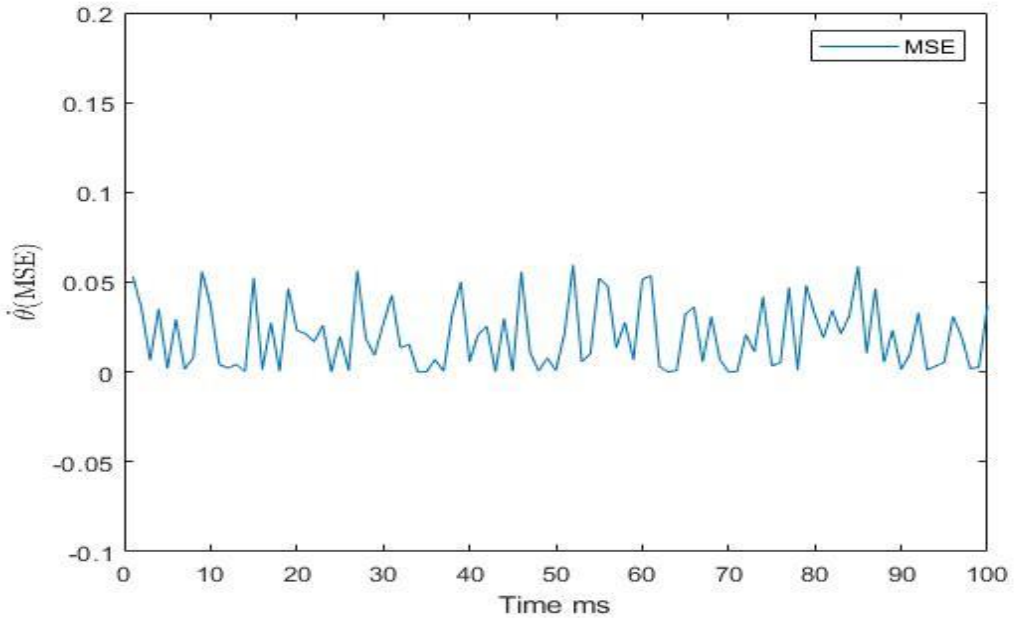

Figure 5. 10: pitch rate controller tracking error

5.3.3. yaw rate controller tracking performance

In Figure 5.11 the yaw rate tracking and from figure 5.12 we can clearly see that mean square error is 0.0081 which shows that even with disturbance the quadcopter traces the reference roll rate efficiently.

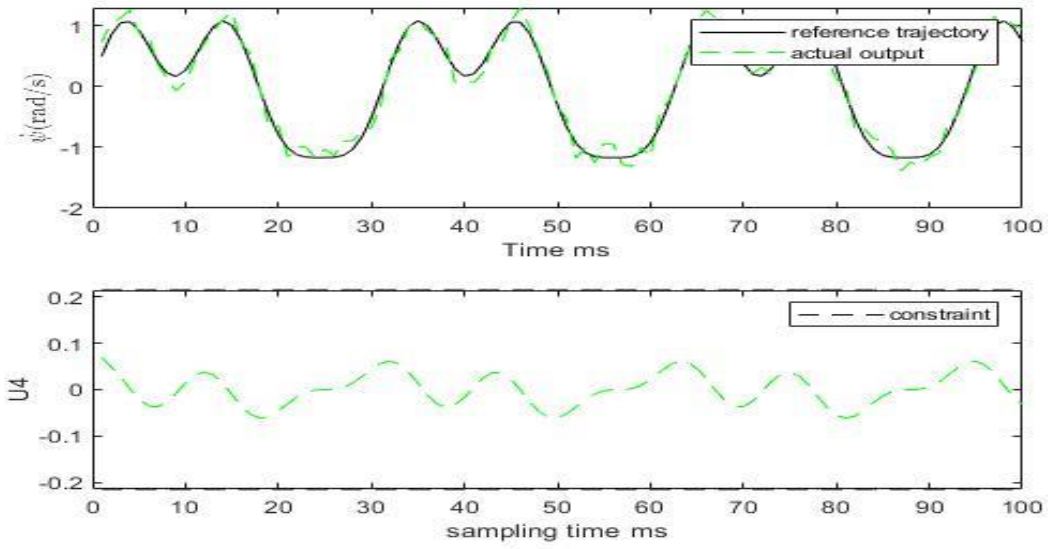

Figure 5. 11: MATLAB yaw rates

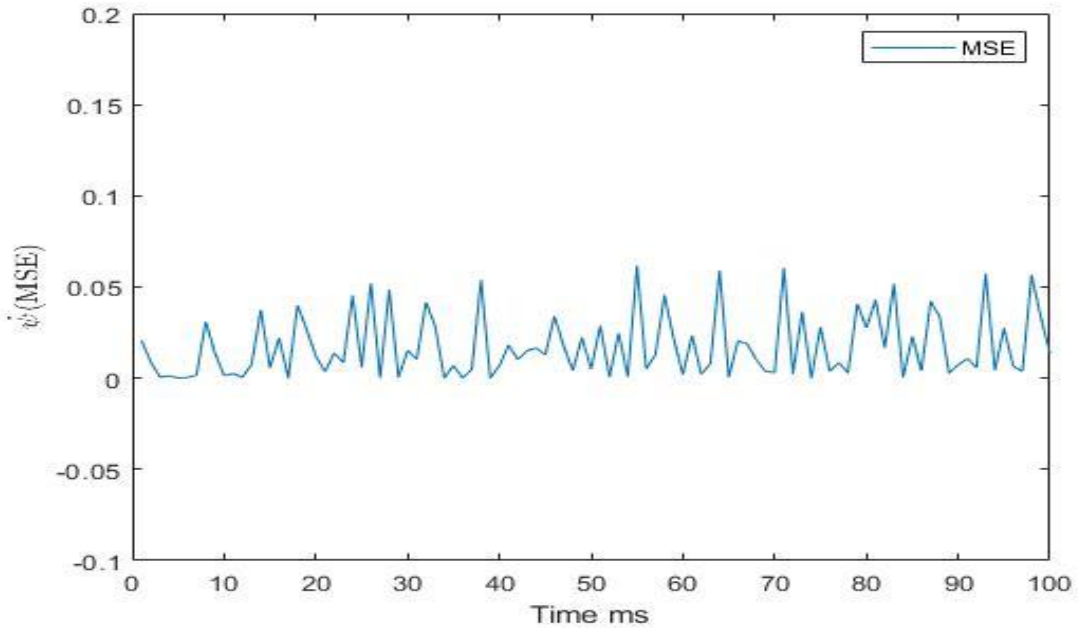

Figure 5. 12: Yaw rate controller tracking error

In table 5.4, a disturbance parameter was adjusted. It is a weighted randomized signal, added to the output vector of the quadcopter model, to replicate the existence of disturbance and sensor noise which present in real world quadrotor flights. It is clearly described in figure from the rates plots that the controller was able to achieve near reference tracking despite the additional disturbance parameter. The dotted horizontal lines in each of the input signal plot represent the constraints that restrict the individual input control signals. By fine-tuning the input weight matrix *W*, promising reference tracking was reached, with the input signals within their respective constraints.

5.4. Trajectory tracking performance with estimator

5.4.1. Position tracking performance without Kalman filter

Figure 5.13, shows that the actual position of the quadcopter tracks the desired trajectories starting from the initial up to the final within 50 seconds of simulation time. Then, the error between the reference trajectory and the actual trajectory is shown clearly. The maximum value of the error is 0.4, which is good.

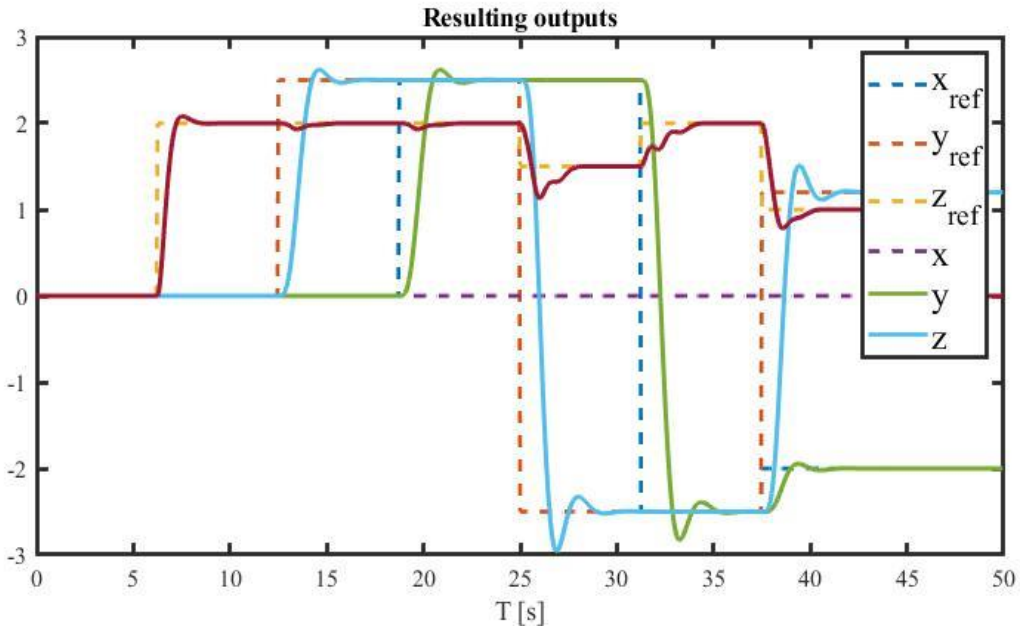

Figure 5. 13: MATLAB position control without Kalman

5.4.2. Position tracking performance with Kalman filter estimator

Figure 5.14, shows that the actual position of the quadcopter tracks the desired trajectories starting from the initial up to the final within 50 seconds of simulation time using state estimator as state input for model predictive controller. Then, the error between the reference trajectory and the actual trajectory is shown clearly. The mean square error of the response is 0.02, which is almost zero. Therefore, the MPC controller with Kalman filter is much better.

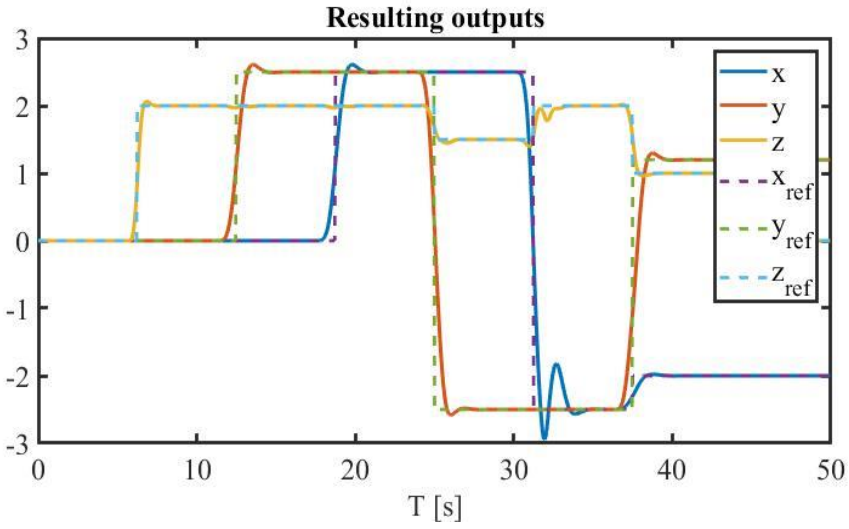

Figure 5. 14: MATLAB position control with Kalman

5.5. Software in the loop gazebo simulation result

The software in the loop physical simulation results were attained by fling the simulation running the improved firmware in Gazebo through tracking waypoints generated by command from terminal SDK and ROS package. The following figure shows that the flight of quadcopter in the gazebo using ROS package and modified px4 firmware and the data is logged using QGroundcontrol from ground control unit which is ubuntu 20.04 virtual environment.

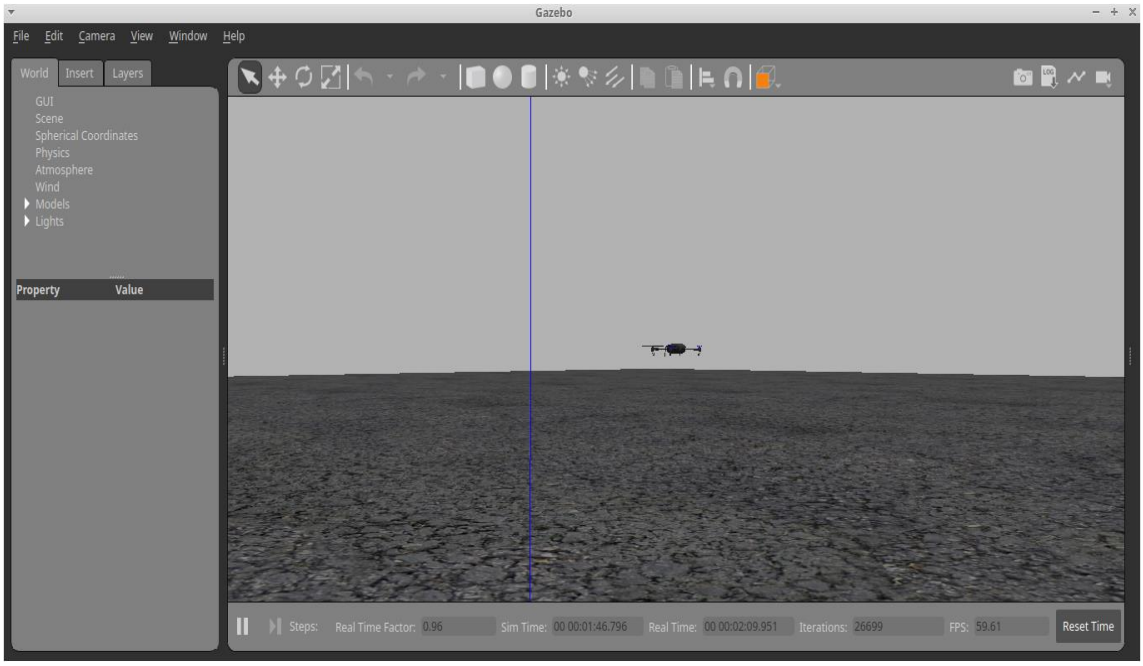

Figure 5. 15: Gazebo flight simulation during flight with z= 2m, hovering

Flight Review is an online data visualization tool, for flight data plotter.

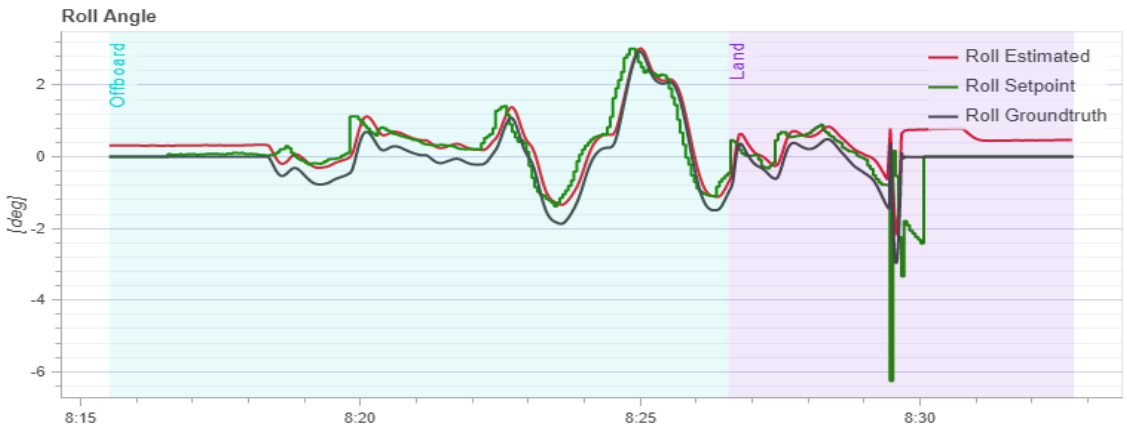

Figure 5. 16: roll angle from SITL simulation

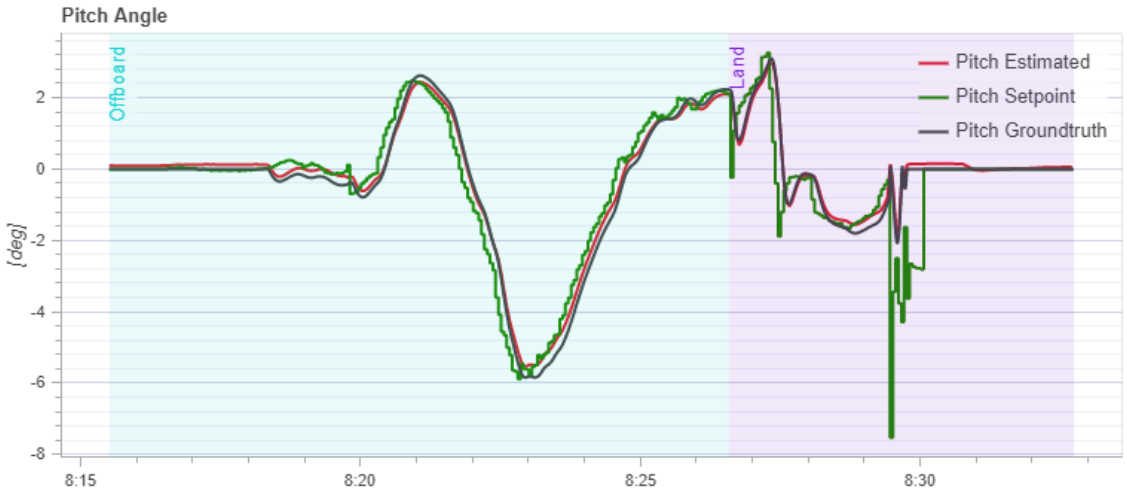

Figure 5. 17: pitch angle from SITL simulation

CHAPTER SIX

6. CONCLUSION AND RECOMMENDATION

6.1. Conclusion

A Quadcopter is a highly nonlinear and unstable system. Without any controller, the response of the plant (drone) is not stable (very rapid increase in output due to small input signal) as shown in the result of stability performance.

In this paper, firstly, a mathematical model of a quadrotor dynamics is derived, using Newton's and Euler's laws. Then a linearized version of the model is obtained, after driving the dynamic model of the system, a Kalman Filter is designed to be used as a state estimator and MPC controller is designed based on the model using Kalman filter and without estimator. To verify the performance and efficiency of the controller, a simulation is obtained using MATLAB and gazebo physical software in the loop simulation. Both MPC and LQEMPC controllers are designed for all output variables. From the result we clearly state that the MPC controller without estimator is not that much effective however LQEMPC is more effective based on performance. Finally, the obtained results show that the system tracks the desired trajectory with small racking error. And the quadcopter back to tracking position even if there would be some external disturbance (wind disturbance). After observing the simulation results, it can be concluded that the quadcopter system is stabilized by controlling the altitudes, position and attitude, of it by using LQEMPC. The developed LQEMPC algorithm possess all the good properties of the MPC while minimizing input variable and additional state estimation.

A future project could be the applying Data Drive Model predictive control technique, like Reinforcement learning (RL) and Moving horizon estimator (MHE), for the real quadrotor to obtain best performance.

6.2. Recommendations

Nowadays, the agricultural sector is a fast-growing economic topic in Africa. The Unmanned aerial vehicle is one of the technologies which is used in many areas of the agriculture. This work is limited to simulation using MATLAB/Simulink. I recommend the hardware implementation of this Quadcopter based on MPC and KF using real-time operating systems like

Texas instruments. implementation, an inertial measurement unit (IMU), which is a combination of an accelerometer and gyroscope measurement, is used to obtain the linear acceleration and angular rate of the drone. Angle from IMU sensor needs to be added fitter like a Kalman filter or complementary filter. On the other hand, on a quadcopter, by adding some future, it is possible to change it to a military purpose by adding cameras and high GPU processor.

7. REFERENCE

Abdelhay, S., & Zakriti, A. (2019). Modeling of a Quadcopter Trajectory Tracking System Using PID Controller. *Procedia Manufacturing*, *32*, 564–571. https://doi.org/10.1016/j.promfg.2019.02.253

Abeywardena, D., Kodagoda, S., Dissanayake, G., & Munasinghe, R. (2013). Improved state estimation in quadrotor MAVs: A novel drift-free velocity estimator. *IEEE Robotics and Automation Magazine*, *20*(4), 32–39. https://doi.org/10.1109/MRA.2012.2225472

Abeywardena, D., Kodagoda, S., Dissanayake, G., & Munasinghe, R. (2015). *Improved State Estimation in Quadrotor MAVs: A Novel Drift-Free Velocity Estimator*. https://doi.org/10.1109/MRA.2012.2225472

Argentim, L. M., Rezende, W. C., Santos, P. E., & Aguiar, R. A. (n.d.). *PID, LQR and LQR-PID on a Quadcopter Platform*. http://www.uastech.com/

Cárdenas R, C. A., Morales, C. A. C., Ospina, J. P., Sánchez, J. F., Caro-Ruiz, C., Grisales, V. H., Ariza-Colpas, P., De-la-Hoz-Franco, E., & González, R. E. R. (2020). Mathematical Modelling and Identification of a Quadrotor. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *12249 LNCS*, 261–275. https://doi.org/10.1007/978-3-030-58799-4_19

Carrillo, L. R. G., Colunga, G. R. F., Sanahuja, G., & Lozano, R. (2014). Quad rotorcraft switching control: An application for the task of path following. *IEEE Transactions on Control Systems Technology*, *22*(4), 1255–1267. https://doi.org/10.1109/TCST.2013.2284790

Chavan, M. S. (2019). Automatic arial vehicle based pesticides spraying system for crops. *International Journal of Innovative Technology and Exploring Engineering*, *8*(11), 41–44. https://doi.org/10.35940/ijitee.J9915.0981119

Collares Pereira, G. (2020). *Lateral Model Predictive Control for Autonomous Heavy-Duty Vehicles*.

Cowling, I. D., Yakimenko, O. A., Whidborne, J. F., & Cooke, A. K. (n.d.). *A Prototype of an*

Autonomous Controller for a Quadrotor UAV.

Dydek, Z. T., Annaswamy, A. M., & Lavretsky, E. (2013). Adaptive Control of Quadrotor UAVs : *IEEE Transactions on Control Systems Technology*, *21*(4), 1400–1406.

Fernando, H. C. T. E., Silva, D., C, D. Z. M. D., & Munasinghe, S. R. (n.d.). *Modelling , Simu ulation and Implemen ntation of a Quadrotor UAV*.

Fessi, R., & Bouallègue, S. (2016). Modeling and Optimal LQG Controller Design for a Quadrotor UAV. *3rd International Conference on Automation, Control, Engineering and Computer Science*, 264–270.

Gao, Y. (2014). Model Predictive Control for Autonomous and Semiautonomous Vehicles. *ProQuest*, *13*(1), 14–17. https://pastel.archives-ouvertes.fr/tel-01635261%0Ahttp://www.ncbi.nlm.nih.gov/pubmed/3640436

Gurbani, V. K., Loreto, S., & Subramanyan, R. (2017). Design and implementation. In *IEEE Communications Magazine* (Vol. 55, Issue 4). https://doi.org/10.1109/MCOM.2017.7901486

Hystad, A. V. (2015). *Model , Design and Control of a Quadcopter Andreas Vikane Hystad*. *May*.

Jiang, Y., Xue, W., Huang, Y., & Fang, H. (2014). The consistent extended Kalman filter. *Proceedings of the 33rd Chinese Control Conference, CCC 2014*, *July*, 6838–6843. https://doi.org/10.1109/ChiCC.2014.6896126

Khan, S., Tufail, M., Khan, M. T., Khan, Z. A., Iqbal, J., & Wasim, A. (2021). Real-time recognition of spraying area for UAV sprayers using a deep learning approach. *PLoS ONE*, *16*(4 April). https://doi.org/10.1371/journal.pone.0249436

Lee, T., & Kang, Y. (2021). Performance analysis of deep neural network controller for autonomous driving learning from a nonlinear model predictive control method. *Electronics (Switzerland)*, *10*(7). https://doi.org/10.3390/electronics10070767

Luukkonen, T. (2012). Modelling and control of quadcopter. *Studies in Systems, Decision and Control*, *24*, 239–255. https://doi.org/10.1007/978-3-319-18290-2_12

MoARD(Ministry of Agriculture and Rural Development). (2010). *Federal Democratic Republic of Ethiopia Ministry of Agriculture and Rural Development*. *2010*(June), 2009–2012.

Mogili, U. R., & Deepak, B. B. V. L. (2018). Review on Application of Drone Systems in Precision Agriculture. *Procedia Computer Science*, *133*, 502–509. https://doi.org/10.1016/j.procs.2018.07.063

Nagaty, A., Saeedi, S., Thibault, C., Seto, M., & Li, H. (2013). Control and navigation framework for quadrotor helicopters. *Journal of Intelligent and Robotic Systems: Theory and Applications*, *70*(1–4), 1–12. https://doi.org/10.1007/s10846-012-9789-z

Njinwoua, B. J., & Wouwer, A. Vande. (2018). Cascade attitude control of a quadcopter in presence of motor asymmetry ∗. *IFAC-PapersOnLine*, *51*(4), 113–118. https://doi.org/10.1016/j.ifacol.2018.06.055

Noordin, A., Mohd Basri, M. A., Mohamed, Z., & Mat Lazim, I. (2021). Adaptive PID Controller Using Sliding Mode Control Approaches for Quadrotor UAV Attitude and Position Stabilization. *Arabian Journal for Science and Engineering*, *46*(2), 963–981. https://doi.org/10.1007/s13369-020-04742-w

Ordaz, P., Espinoza, E. S., Muñoz, F., Carrillo, L. R. G., Romero, H., & Lozano, R. (2018). *Nonlinear Control and Trajectory Tracking of an Unmanned Aircraft System Based on a Complete State Space Representation*. *51*(13), 561–566. https://doi.org/10.1016/j.ifacol.2018.07.339

Praveenkumar, U., & Prabhakaram, G. (2014). Design and Fabrication of High Endurance Ruav With Rechargeable Solar Power Source. *International Journal of Mechanical And Production Engineering*, *2*(2), 63–69.

Robinson, P. R., & Cima, D. (2017). Model-predictive control fundamentals. *Springer Handbooks*, *PartF1*(2), 833–839. https://doi.org/10.1007/978-3-319-49347-3_26

Rq, L. D., Mxvw, X., Fq, H. G. X., Kdyh, U., Rewdlqhg, E., Wkh, L. Q., Wudfnlqj, W., Wkh, R. I., Lv, T., Zlqg, L. E., Lqwhuqdo, J., & Dqg, I. (n.d.).

Valavanis, K. P., & Vachtsevanos, G. J. (2015). Handbook of unmanned aerial vehicles. In *Handbook of Unmanned Aerial Vehicles*. Springer Netherlands.

https://doi.org/10.1007/978-90-481-9707-1

Venkatesh, G. A., Sumanth, P., & Jansi, K. R. (2017). Fully autonomous UAV. *Proceedings - 2017 International Conference on Technical Advancements in Computers and Communication, ICTACC 2017*, *2017-October*, 41–44. https://doi.org/10.1109/ICTACC.2017.20

Xuan-Mung, N., & Hong, S. K. (2019). Improved altitude control algorithm for quadcopter unmanned aerial vehicles. *Applied Sciences (Switzerland)*, *9*(10). https://doi.org/10.3390/app9102122

Zhang, X., Li, X., Wang, K., & Lu, Y. (2014). A survey of modelling and identification of quadrotor robot. *Abstract and Applied Analysis*, *2014*. https://doi.org/10.1155/2014/320526

APPENDICES

A. MATLAB script for MPC design

```
%%% the
 % This program prepares the quadcopter parameters and matrices needed to
 % run the MPC control program.

%% Parameter Declarations

tic % initialize timer

% Quadcopter parameters

 m = 1.587; % Mass of the quadcopter [kg]
 g = 9.81; % Accelleration of  gravity [m/(s^2)]
 Ixx = 0.0213; % moment of inertia in x-axis [kg*m^2]
 Iyy = 0.02217; % y
 Izz = 0.0282; % z
 K = 4.0687e-7; % Thrust constant [N/(rpm)^2]
 b = 8.4367e-9; % Drag constant [N*m/(rpm)^2]
 d_arm = 0.243; %
 max_speed = 7760; % Maximum [rpm]
 min_speed = 6093; % Minimum [rpm]

 % Controller design parameters

 no_of_states = 6;
 no_of_inputs = 3;
 no_of_outputs = 3;
 nu = 3;
 ny = 6;
 N_sim = 20;
 x = zeros(6,1); % Initial state
 y = zeros(3,1); % Initial output
 u = [0;0;0]; % Initial control
 t = 0:0.2:N_sim; % Steps for sinusoidal ref signal

 %  % Reference to be tracked
 %  r1 = heaviside(t);
 %  r2 = heaviside(t);
 %  r3 = heaviside(t);
 %  r = [r1;r2;r3];
 r = [sin(t) + cos(3*t)/2; sin(t) + cos(3*t)/2; sin(t) + cos(2*t)/2 +
sin(3*t)/3];
```

```
  Rs = ref_adj(no_of_outputs, ny); % Reference Adjusting Matrix
  %dist = (zeros(3,101)*2 -0.0000);
  dist = (rand(3,101)*2 -1)*0.25; % Disturbance

  %% State Space Matrices

  A = [0 0 0 1 0 0 ;...
       0 0 0 0 1 0 ;...
       0 0 0 0 0 1;...
       0 0 0 0 0 0;...
       0 0 0 0 0 0 ;...
       0 0 0 0 0 0];

  B = [0     0     0;
       0     0     0;
       0     0     0;
       1/Ixx 0     0;
       0     1/Iyy 0;
       0     0     1/Izz];

  C = [0 0 0 1 0 0;...
       0 0 0 0 1 0;...
       0 0 0 0 0 1];

  D = zeros(3,3);

  %% Converting from Continuous to Discrete Time

  [Ad,Bd,Cd,Dd] = c2dm(A,B,C,D,0.2);

 %% Augmented Model

  [A_aug,B_aug,C_aug] =
augment_mimo(Ad,Bd,Cd,no_of_states,no_of_inputs,no_of_outputs);
  % Controllability check
  CO = [B_aug A_aug*B_aug A_aug^2*B_aug A_aug^3*B_aug A_aug^4*B_aug
A_aug^5*B_aug];

  rank(CO);


  %% Prediction Matrices

  [H,P] = mpc_predictions_output(A_aug,B_aug,C_aug,ny,nu); % Output Predictions
```

```
%% Constraints

 % Constraint calculations

 u_1 = d_arm * b*(max_speed^2 - min_speed^2);
 u_2 = u_1;
 u_3 = k * (max_speed^2 + max_speed^2 - min_speed^2 - min_speed^2);

 umax = [u_1; u_2; u_3];
 umin = -umax;
 Dumax = 0.6*umax;

 % Constraint matrices

 [CC, dd, dupast] = constraints_mimo(Dumax,umax,umin,no_of_inputs,nu);

 %% Cost Function & its Parameters

 % Weights on control inputs



 W = [7.5e-2 0 0 0 0 0 0 0 0 ;...
 0 7.5e-2 0 0 0 0 0 0 0 ;...
 0 0 4.5e-2 0 0 0 0 0 0;...
 0 0 0 7.5e-2 0 0 0 0 0 ;...
 0 0 0 0 7.5e-2 0 0 0 0 ;...
 0 0 0 0 0 4.5e-2 0 0 0 ;...
 0 0 0 0 0 0 7.5e-2 0 0 ;...
 0 0 0 0 0 0 0 7.5e-2 0 ;...
 0 0 0 0 0 0 0 0 4.5e-2];
 E = 2*(H'*H + W);


 % Simulation loop parameters

 Xf = zeros(size(B_aug,1),1);
 xh = zeros(size(B_aug,1),1);
 yh = y;

 % Simulation loop

 for i = 1:100

 F = -2*H'*(Rs*r(:,i) - P*(Xf));
```

```
 d = dd + dupast*u;

 % Quadratic Progamming

 [DeltaU, Uncons] = QPhild(E,F,CC,d);

 % For control horizon of 2
 % DeltaU = [DeltaU(1,1) DeltaU(2,1) DeltaU(3,1);...
 % DeltaU(4,1) DeltaU(5,1) DeltaU(6,1)];

 % For control horizon of 3
 DeltaU = [DeltaU(1,1) DeltaU(2,1) DeltaU(3,1);...
 DeltaU(4,1) DeltaU(5,1) DeltaU(6,1);...
 DeltaU(7,1) DeltaU(8,1) DeltaU(9,1)];

 deltau = DeltaU(1,:);
 u = u + deltau'; % Input

 % Model

 xh(:,i+1) = A_aug*xh(:,i) + B_aug*deltau';
 yh(:,i) = C_aug*xh(:,i+1) + dist(:,i);
 Xf = xh(:,i+1);


 % % Same result with model above
 % x(:,i+1) = Ad*x(:,i) + Bd*u;
 % y(:,i) = Cd*x(:,i+1) + dist(:,i);
 % Xf = [x(:,kk+1) - x(:,kk); y(:,kk)]

 % Plot variables

 u1(:,i) = u;
 delt(:,i) = deltau;

 % Scaled input
 % u_s = (2 * ((u(:,1) - umin(:,1))./(umax(:,1) - umin(:,1)))- 1);

 end

 toc % terminate timer

%% Plots

 % i=1:N_sim;
 i = 1:100;
```

```
  % Output 1
  figure (1)
  plot(i,r(1,i),'k',i,yh(1,i),'--g')
  legend('reference trajectory','actual output')
  xlabel('Time {ms}')
  ylabel('$\dot{\phi}$(rad/s)','Interpreter','latex')

  % Output 2
  figure (2)
  plot(i,r(2,i),'k',i,yh(2,i),'--g')
  legend('reference trajectory','actual output')
  xlabel('Time {ms}')
  ylabel('$\dot{\theta}$(rad/s)','Interpreter','latex')

  % Output 3
  figure (3)
  plot(i,r(3,i),'k',i,yh(3,i),'--g')
  legend('reference trajectory','actual output')
  xlabel('Time {ms}')
  ylabel('$\dot{\psi}$(rad/s)','Interpreter','latex')

  figure (4)
  subplot(3,1,1)
  plot(i, umax(1)*ones(1,100), '--k', i, umin(1)*ones(1,100), '--k',i, u1(1,i),
'--g')
  title('Plots of input over a sample period')

  subplot(3,1,2)
  plot(i, umax(2)*ones(1,100), '--k', i, umin(2)*ones(1,100), '--k',i, u1(2,i),
'--g')

  subplot(3,1,3)
  plot(i, umax(3)*ones(1,100), '--k', i, umin(3)*ones(1,100), '--k',i, u1(3,i),
'--g')

  figure (5)
  subplot(3,1,1)
  plot(i, Dumax(1)*ones(1,100), '--k', i, -Dumax(1)*ones(1,100), '--k', i,
delt(1,i), '--g')
  title('Change in input over a sample period')

  subplot(3,1,2)
  plot(i, Dumax(2)*ones(1,100), '--k', i, -Dumax(2)*ones(1,100), '--k', i,
delt(2,i), '--g')
```

```
  subplot(3,1,3)
  plot(i, Dumax(3)*ones(1,100), '--k', i, -Dumax(3)*ones(1,100), '--k', i,
delt(3,i), '--g')
```

B. MATLAB script for MPC predictions output

```
% H and P are the output prediction matrices

function [H, P] = mpc_predictions_output(A,B,C,ny,nu)

prod = C*A;
size_prod = size(prod);

% P matrix
P = [];
j = 1; % P matrix index
for i=1:ny
    P(j:i*size_prod(1),1:size_prod(2)) = C*(A^(i));
    j = j+size_prod(1);
end

% H matrix
prod2 = C*A*B;
size_prod2 = size(prod2);
k = 1; % Row element variable
l = 1 + size_prod2(2); % Column element variable

for i=1:ny % for row elements
    H(k:i*size_prod2(1),1:size_prod2(2)) = C*A^(i-1)*B;
    if_var = i - 1; % Variable for if condition
    for t=1:nu-1 % Column elements
        if(if_var) <= 0
            H(k:i*size_prod2(1),l:l + size_prod2(2)-1) =
zeros(size_prod2(1),size_prod2(2));
        else
            H(k:i*size_prod2(1),l:l + size_prod2(2)-1) = C*A^(i-1-t)*B;
        end
            if_var = if_var - 1;
            l = l + size_prod2(2);
    end
    k = k + size_prod2(1);
    l = 1 + size_prod2(2); % To reset the column index element
end
```

C. Controllability, observability and stability

```
%% to check controllability , observability and stability
% dx = Ax + Bu
% y = Cx + Du
% lqr(Q,R)
clear all; clc;
close all;
m = 2;
g = 9.8;
Ixx = 0.22;
Iyy = 0.22;
Izz = 0.22;

 A = [  0     0     0     1     0     0     0       0     0     0     0
0;...

        0     0     0     0     1     0     0       0     0     0     0
0;...
        0     0     0     0     0     1     0       0     0     0     0
0;...
        0     0     0     0     0     0     0       g     0     0     0
0;...
        0     0     0     0     0     0    -g       0     0     0     0
0;...
        0     0     0     0     0     0     0       0     0     0     0
0;...
        0     0     0     0     0     0     0       0     0     1     0
0;...
        0     0     0     0     0     0     0       0     0     0     1
0;...
        0     0     0     0     0     0     0       0     0     0     0
1;...
        0     0     0     0     0     0     0       0     0     0     0
0;...
        0     0     0     0     0     0     0       0     0     0     0
0;...
        0     0     0     0     0     0     0       0     0     0     0     0
];
 B =   [    0     0     0         0
            0     0     0         0
            0     0     0         0
            0     0     0         0
            0     0     0         0
          1/m     0     0         0
```

```
            0     0     0         0
            0     0     0         0
            0     0     0         0
            0     1/Ixx     0     0
            0     0     1/Iyy     0
            0     0     0     1/Izz ];

 C =  [       1     0     0     0     0     0     0     0     0     0     0
0
             0     1     0     0     0     0     0     0     0     0     0
0
             0     0     1     0     0     0     0     0     0     0     0
0
             0     0     0     0     0     0     1     0     0     0     0
0
             0     0     0     0     0     0     0     1     0     0     0
0
             0     0     0     0     0     0     0     0     1     0     0
0
     ];
 D = zeros(6,4);
 states = {'X' 'Y' 'Z' 'dX' 'dY' 'dZ' 'phi' 'theta' 'psi' 'p' 'q' 'r'};
 inputs = {'throttel' 'roll' 'pich' 'yaw'};
 outputs = {'X' 'Y' 'Z' 'phi' 'theta' 'psi'};
 sys_mimo = ss(A,B,C,D, 'statename' ,states,...
     'inputname' ,inputs,...
     'outputname' ,outputs);
 % j = tf(sys_mimo);
 % j(3,1);
 % step(sys_mimo);
 %% controllability and observability
 CO= ctrb(A,B);
 rc= rank(CO);
 OB= obsv(A,C);
 ro= rank(OB);
```