Addis Ababa University

Addis Ababa institute of Technology

School of Electrical and Computer Engineering

Trajectory Tracking Control of ISR Quadrotor UAV using GA Optimized Fuzzy-PID Based Neural Network Controller

---

By

Nigatu Wanore

A Thesis Submitted to School of Graduate Studies of Addis Ababa University in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Control Engineering

Thesis Advisor

Lebsework Negash(PhD)

November 16, 2023

Addis Ababa, Ethiopia

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

This is to certify that the thesis prepared by Nigatu Wanore entitled "Trajectory Tracking Control of ISR Quadrotor UAV using GA Optimized Fuzzy-PID Based Neural Network Controller" submitted in partial fulfillment of the requirements for the degree of Master of Sciences in Control Engineering complies with the regulations of university and meets the accepted standards with respect to originality and quality.

Approved by Board of Examiners

Name	Signature	Date
Bisrat Derebssa(PhD)		
(School dean)		
Lebsework Negash(PhD)		
(Advisor & Control Chair)		
Mr.Mihretab Negash		
(Internal Examiner)		
Chala Merga(PhD)		
(External Examiner)		

Declaration

I the undersigned, certify that this research work entitled "Trajectory Tracking Control of ISR Quadrotor UAV using GA Optimized Fuzzy-PID Based Neural Network Controller" is my original work. The work has not been presented for the fulfillment of any degree in this or any other University where all sources and materials have been used for the thesis from other sources are properly acknowledged.

Nigatu Wanore		November 16 , 2023
Name of Student	Signature	Submission Date

This thesis has been submitted for examination with my approval as a University advisor.

Name of Advisor	Signature
Lebsework Negash(PhD)	

Dedication

To my wife,Bethlehem Tadesse, for her unwavering love, support, and understanding.

Acknowledgment

Firstly, I would like to express my immense gratitude to the Almighty God,for unwavering support, which has provided me with stability and strength during challenging times.

I would like to extend my sincerest appreciation to my advisor, Lebsework Negash(PhD), for generously sharing his expertise, offering guidance, encouragement, and providing valuable insights throughout this endeavor. Without his guidance, this undertaking would not have been possible. It has been a remarkable privilege to have such an exceptional instructor and scholar as my mentor, from whom I have gained inspiration and profound respect.

I am also deeply thankful to the Information Network Security Administration (INSA) for sponsoring my studies in the Master of Science in Control Engineering program at AAiT School of Electrical and Computer Engineering. Their support has been instrumental in enabling me to pursue my educational goals.

Nigatu Wanore

Abstract

The ISR Quad-Rotor is an unmanned aerial vehicle (UAV) utilized for intelligence gathering, surveillance, and reconnaissance missions. This thesis focuses on modeling and controlling the quad-rotor to identify and track a Person of interest using an intelligent control technique. The initial step involves deriving a nonlinear mathematical model for the 6DOF (six degrees of freedom) Quadrotor UAV using the Newton-Euler formalism. To ensure trajectory tracking of the quad-rotor, a Fuzzy Proportional-Integral-Derivative (Fuzzy-PID) controller, tuned with a Genetic Algorithm (GA), was implemented to generate data for a neural network (NNFPID-GA). The fuzzy logic approach facilitates parameter adjustment based on predefined fuzzy rules, while the GA algorithm determines the scaling factors of the Fuzzy-PID controller. The proposed control system was designed based on input-output data from the GA Optimized Fuzzy-PID controller. A network was trained using the Levenberg-Marquardt backpropagation algorithm with the assistance of the MATLAB®NN Toolbox. MATLAB® simulations were conducted to validate the effectiveness of the proposed control algorithm. Additionally, a flight test were performed using the UAV Toolbox to assess the stable flight performance of a developed GA Optimized Fuzzy-PID based neural network(NNFPID-GA) controller for the Quadrotor UAV. To evaluate the performance of both controllers, a comparison study based on performance metrics was conducted. Even though both controllers offer faster response in terms of settling time and improved performance, with less overshoot and better robustness in handling parameter variation and disturbance rejection capability, the GA Optimized Fuzzy-PID based neural network controller(NNFPID-GA) outperforms the GA Optimized Fuzzy-PID controller(FPID-GA). Finally, in order to generate trajectories for the controller, a face recognition system using Python were implemented. The training results demonstrate an accuracy of 98.65%, indicating that the system can effectively distinguish between a wanted person (Person of interest) and other individuals.

Keywords– Quad-Rotor, Neural Network, FPID-GA,NNFPID-GA, Genetic Algorithm, Degrees of Freedom, ISR, UAV.

Contents

Declaration i

Acknowledgment iii

List of Acronyms xii

1 Introduction 1
- 1.1 Background . . . 1
- 1.2 Mission Definition . . . 2
- 1.3 Problem Statement . . . 2
- 1.4 Objectives . . . 2
  - 1.4.1 General Objective . . . 2
  - 1.4.2 Specific Objectives . . . 2
- 1.5 Significance of the Research Work . . . 3
- 1.6 Methodology . . . 3
- 1.7 Scope of the Thesis . . . 4
- 1.8 Outlines of the Thesis . . . 5

2 Literature Review 6
- 2.1 Overview of ISR Quadrotor and It's Importance in ISR Application . . . 6
- 2.2 Face Recognition and Identification . . . 7
- 2.3 Control of Quadrotor . . . 8

3 Modelling of Quad-Rotor UAV 11
- 3.1 Modeling Approach . . . 11
  - 3.1.1 Types of Configuration of Quad-Rotor . . . 11
- 3.2 Reference Frame Alignment . . . 11
  - 3.2.1 Body-fixed Reference Frame(BFF) . . . 13
  - 3.2.2 Earth or Inertial Reference Frame(IFF) . . . 13
  - 3.2.3 Assumptions for Quad-Rotor Modelling and its Operation Principles 13
    - 3.2.3.1 Operation Principle of Quad-Rotor . . . 14
  - 3.2.4 Quad-Rotor Dynamics . . . 15
    - 3.2.4.1 Basic Forces Applied to Quad Rotor Body . . . 16
  - 3.2.5 Kinematics for Quad-Rotor . . . 16
- 3.3 Model Verification . . . 26
  - 3.3.1 Hovering Verification . . . 27
  - 3.3.2 Altitude Verification(Throttle up) . . . 28

3.3.3 Roll Verification . . . 28
3.3.4 Pitch Verification . . . 29
3.3.5 Yaw Verification . . . 30

4 Controller Design for Quad-Rotor 31
4.1 Introduction . . . 31
4.2 GA Optimized Fuzzy-PID Controller Design . . . 32
4.2.1 Fuzzy System for Tuning the PID Gains Online . . . 32
4.2.2 Fuzzy-PID/PD Gain Tuning . . . 36
4.2.3 Genetic Algorithm Optimization (GA) . . . 37
4.2.3.1 Fitness Function . . . 38
4.3 GA Optimized Fuzzy-PID Based Neural Network Controller(NNFPID-GA) Design . . . 40
4.3.1 Introduction . . . 40
4.3.2 Basic Neural Network Model . . . 42
4.3.2.1 Levenberg Marquardt Algorithm-Based Backpropagation Learning Algorithm . . . 44
4.3.2.2 Advantages of NN Controller . . . 44
4.3.2.3 Network Architecture . . . 46
4.4 Fixed point Tracking(Step Response)Control . . . 46
4.5 Position and Altitude controller . . . 46
4.5.1 Position x Controller . . . 47
4.5.2 Position y Controller . . . 47
4.5.3 Altitude z Controller . . . 48
4.6 Attitude and Heading controller . . . 48
4.6.1 Heading Yaw (ψ) controller . . . 48
4.6.2 Attitude Roll(ϕ)controller . . . 48
4.6.3 Attitude Pitch(θ)controller . . . 48
4.7 Trajectory Tracking Control . . . 49
4.8 Rectangular Trajectory Tracking Control . . . 50
4.8.1 Position x Controller . . . 50
4.8.2 Position y Controller . . . 51
4.8.3 Altitude z Controller . . . 52
4.8.4 Heading Yaw(ψ)Controller . . . 52
4.8.5 Attitude Roll(ϕ)controller . . . 53
4.8.6 Attitude Pitch(θ)Controller . . . 53
4.9 Square Wave Trajectory Tracking Control . . . 54
4.10 Infinity Trajectory Tracking Control . . . 54
4.11 Circular Trajectory Tracking Control . . . 54
4.12 Spiral Trajectory Tracking Control . . . 55

5 Face Recognition and Tracking with ISR Quad-Rotor 56
5.1 Introduction . . . 56
5.2 Face Recognition with Python . . . 56
5.2.1 Open CV . . . 57
5.3 Face Recognition and Tracking Algorithm . . . 58

6 Simulation Results and Discussion **61**
6.1 Introduction . . . 61
6.2 Rectangular Trajectory Tracking . . . 62
6.3 Triangular Trajectory Tracking . . . 63
6.4 Square Wave Trajectory Tracking . . . 65
6.5 Circular Trajectory Tracking . . . 66
6.6 Infinity Trajectory Tracking . . . 67
6.7 Spiral Trajectory Tracking . . . 69
6.8 Flight test using UAV Toolbox . . . 70
6.9 Step Response . . . 71
6.10 Parameter Variation and Disturbance Rejection Capability . . . 72
6.11 Rectangular Trajectory Tracking with Disturbance . . . 72
6.12 Rectangular Trajectory Tracking with Parameter Variation . . . 74
6.13 Rectangular Trajectory Tracking with Disturbance and Parameter Variation 75
6.14 Step Response with Disturbance and Parameter Variation . . . 77
6.15 Comparison of Controllers Based on Performance Indices(Measurement) . . . 78
6.16 Comparison of Controllers Based on Performance Indices Using Trajectory Tracking Performance . . . 78
6.16.1 Comparison of Controllers Based on Performance Indices Using Rectangular Trajectory Tracking Performance . . . 78

7 Conclusion and Recommendation for Future Work **82**
7.1 Conclusion . . . 82
7.2 Recommendation for Future Work . . . 83

A Complete MATLAB®SIMULINK Plant with NNFPID-GA& GA Optimized Fuzzy-PID Controller **88**
A.1 A Single DOF Control with GA Optimized Fuzzy-PID & NNFPID-GA Controller . . . 89
A.2 Performance Indices . . . 91

B Pseudo Code For Square Wave Trajectory **92**

C **94**
C.1 Genetic Algorithm to tune Scaling Factors . . . 94
C.2 Fitness Function for GA . . . 94
C.3 Function Main for Face Recognition . . . 94
C.4 Phyton Code for Face Recognition . . . 94

List of Figures

3.1 a)cross (x)configuration,b)plus(+)configuration 11
3.2 Reference Frame 13
3.3 Working Principle of Quad-rotor [26] 16
3.4 Basic Forces Applied to Quad-rotor [26] 16
3.5 Model of Quad-Rotor Dynamics 26
3.6 Complete Model of Quad-Rotor Dynamics 26
3.7 Hovering Verification 28
3.8 Drone Rise up Altitude Verification 28
3.9 Drone Fall down Altitude Verification 29
3.10 Roll Verification 29
3.11 Pitch Verification 30
3.12 Yaw Verification 30

4.1 Control scheme Block Diagram 33
4.2 Basic Structure of Plant(ISR Quadrotor) with Fuzzy-PID 33
4.3 Structure of GA Optimized Fuzzy-PID Controller 34
4.4 Fuzzy-PID 2 Inputs 3 Outputs and 49 rules 36
4.5 Inputs error and its rate Membership Function 37
4.6 Output k_p and k_i Membership Function 37
4.7 Output k_d Membership Function and over all Fuzzy-PID Rule Viewer 38
4.8 Flow Chart for GA Optimization 39
4.9 The Relationship between Biological Neural Network and Artificial Neural Network [41] 41
4.10 Feedforward Neural Network Architecture 42
4.11 Work Flow of NN 43
4.12 Jobs of Neural Network(Computation Algorithm) 43
4.13 Backpropagation of errors(Computation Algorithm) 45
4.14 NN Control 45
4.15 The Internal structure of Proposed Controller(NNFPID) 46
4.16 Training Result for Position x Controller after 12 Iterations 47
4.17 Training Result for Position y Controller after 109 Iterations 47
4.18 Training Result for Altitude z Controller after 417 Iterations 48
4.19 Training Result for Heading(ψ) Controller after 8 Iterations 49
4.20 Training Result for Roll(ϕ)Controller after 8 Iterations 49
4.21 Training Result for Pitch(θ) Controller after 9 Iterations 50
4.22 Application based Mission Scenario 51
4.23 Training Result for Position x Controller after 33 Iterations 51

4.24 Training Result for Position y Controller after36 Iterations 52
4.25 Training Result for Altitude z Controller after 10 Iterations 52
4.26 Training Result for Heading Yaw(ψ)Controller after 11 Iterations 53
4.27 Training Result for Attitude Roll(ϕ)Controller after 9 Iterations 53
4.28 Training Result for Attitude Pitch(θ)Controller after 7 Iterations 54

5.1 Flow Chart for Face Recognition and Identification Phases 58
5.2 Face Recognition and Identification Phases 58
5.3 Training Result at different Face Conditions 59
5.4 Training accuracy Identification/Recognition of Wanted Person (Person of interest) from Others 59
5.5 Complete ISR Mission using Quad-Rotor (Operational Scenario) 60
5.6 Block Diagram for Autonomous Face Recognition and Tracking by ISR Quad-Rotor 60

6.1 Control efforts for Rectangular Trajectory Tracking 62
6.2 Position Tracking Response for Rectangular Trajectory Tracking 62
6.3 3D Plot for Rectangular Trajectory Tracking 63
6.4 Person of interest tracking using ISR Quadrotor 63
6.5 Control efforts for Triangular Trajectory Tracking 64
6.6 Position Tracking Response for Triangular Trajectory Tracking 64
6.7 3D Plot for Triangular Trajectory Tracking 64
6.8 Control efforts for Square wave Trajectory Tracking 65
6.9 Position tracking Response for Square wave Trajectory Tracking 65
6.10 3D Plot for for Square wave Trajectory Tracking 66
6.11 Control efforts for Circular Trajectory Tracking 66
6.12 Position Tracking Response for Circular Trajectory Tracking 67
6.13 3D Plot for Circular Trajectory Tracking 67
6.14 Control efforts for Infinity Trajectory Tracking 68
6.15 Position Tracking Response for Infinity Trajectory Tracking 68
6.16 3D Plot for Infinity Trajectory Tracking 68
6.17 Control efforts for Spiral Trajectory Tracking 69
6.18 Position Tracking Response for Spiral Trajectory Tracking 69
6.19 3D Plot for Spiral Trajectory Tracking 70
6.20 Different types of Flight test under GA Optimized Fuzzy-PID Based Neural Network Controller using UAV Toolbox 71
6.21 Control efforts for Step Response 71
6.22 Position Response for Step Response 72
6.23 The Input Disturbance added along Positions for Rectangular Trajectory Tracking of Quad-Rotor 72
6.24 Control efforts for Rectangular Trajectory with added Input Disturbance . . 73
6.25 Position Tracking Response for Rectangular Trajectory with added Input Disturbance 73
6.26 3D Plot for Rectangular Trajectory with added Input Disturbance 73
6.27 Control efforts for Rectangular Trajectory with Parameter Variation 74
6.28 Position Tracking Response for Rectangular Trajectory with Parameter Variation 74

6.29 3D Plot for Rectangular Trajectory with Parameter Variation 75
6.30 Control efforts for Rectangular Trajectory with added Input Disturbance and Parameter Variation 75
6.31 Position Tracking Response for Rectangular Trajectory with added Input Disturbance and Parameter Variation 76
6.32 3D Plot for Rectangular Trajectory with added Input Disturbance and Parameter Variation 76
6.33 (Person of interest tracking where yellow colour indicates person of interest and blue indicates Quadrotor Trajectory)3D Plot for Rectangular Trajectory with added Input Disturbance and Parameter Variation 76
6.34 The Input Disturbance added along Positions for Step Response 77
6.35 Control efforts for Step Response with added Input Disturbance and Parameter Variation 77
6.36 Position Tracking Response for Step Response with added Input Disturbance and Parameter Variation 77

A.1 Complete Plant Model with Controllers 88
A.2 Altitude Control using GA Optimized Fuzzy-PID Controller 89
A.3 Input Output Training Data for NNFPID-GA Controller 89
A.4 Altitude Control using NNFPID-GA Controller based on I/O training Data from GA Optimized Fuzzy-PID Controller 90
A.5 Altitude Control(NNFPID-GA)Internal Structure 90

List of Tables

3.1 Parameters Description. 27
3.2 Parameters Values [30] 27

4.1 Fuzzy rules for K_d [34] 35
4.2 Fuzzy rules for K_p andK_i [34] 36

6.1 Controllers Performance Indices for Rectangular Trajectory 78
6.2 Controllers Performance Indices for Rectangular Trajectory Tracking with Input Disturbance 79
6.3 Controllers Performance Indices for Rectangular Trajectory Tracking with Parameter Variation 80
6.4 Controllers Performance Indices for Rectangular Trajectory Tracking with Input Disturbance and Parameter Variation 81

List of Acronyms

AI	Artificial Intelligence
CCW	Counter Clock wise
CW	Clock wise
CCTV	Closed-Circuit Television
Fuzzy-PID	Fuzzy tuned Proportional Integral Derivative controller
GA Optimized Fuzzy-PID(FPID-GA)	Fuzzy tuned Proportional Integral Derivative controller Scaling factors tuned with Genetic Algorithm
GA	Genetic Algorithm
INSA	Information Network Security Administration
ISR	Intelligence, Surveillance and Reconnaissance
LiDAR	Light Detection and Ranging
LQR	Linear Quadratic Regulator
MFs	Membership Functions
NNFPID-GA	GA Optimized Fuzzy-PID Based Neural Network Controller
NN	Neural Network
OpenCV	Open Source Computer Vision Library
PID	Proportional, Integrator and Derivative
SMC	Sliding Mode Control
UAS	Unmanned Aerial System
UAV	Unmanned Aerial Vehicle

Chapter 1

Introduction

1.1 Background

ISR (Intelligence, Surveillance, and Reconnaissance) Quadrotors are unmanned aerial vehicles (UAVs) specifically designed for gathering intelligence, conducting surveillance missions, and performing reconnaissance in various applications.The use of ISR Quadrotors has gained significant importance in fields such as military operations, law enforcement, disaster manage ment, and environmental monitoring. These small, agile, and maneuverable quadrotors can access confined spaces, hover, and navigate in complex environments, making them ideal for ISR purposes.The primary objective of an ISR Quadrotor is to gather data and assess the situation in real-time, providing valuable information for decision-making processes. They are equipped with various sensors, including high-resolution cameras, thermal imaging sensors, LiDAR (Light Detection and Ranging), and other specialized equipment, depending on the specific mission requirements.

The ISR Quadrotors can perform a wide range of tasks, such as:

Surveillance and Reconnaissance: They provide real-time video footage, aerial imagery, and other sensor data to monitor areas of interest, identify potential threats, and gather intelligence on enemy activities or critical situations.

Target Tracking and Monitoring: They can track and monitor moving targets, such as vehicles, individuals, or objects, providing continuous surveillance and valuable data for situational awareness.

Disaster Management: ISR Quadrotors play a vital role in disaster management by assessing the extent of damage, mapping affected areas, and helping rescue teams with search and rescue operations.

Environmental Monitoring: They enable efficient monitoring of environmental factors such as air quality, temperature, humidity, and vegetation mapping, aiding in resource management and environmental research.

Border Security: ISR Quadrotors provide border patrol agencies with enhanced surveilla nce capabilities, enabling timely detection of unauthorized border crossings, drug trafficking, or other illegal activities.

1.2 Mission Definition

The mission aims to enhance intelligence, surveillance, and reconnaissance (ISR) capabilities in military and law enforcement operations, in order to protect the national interests of the Country. By utilizing drone technology equipped with integrated AI face recognition, the mission intends to track terrorist groups and wanted individuals, assisting decision-makers in making informed decisions while upholding the rule of law in major cities. The proposed Quadrotor drone will be small in size and programmed in Python to execute AI face identification. To track the wanted person, the drone's camera will follow their facial positions based on the x and y coordinates provided, autonomously ensuring law enforcement personnel obey the rule of law. To enable stable control and identification, an intelligent controller employing a GA optimized fuzzy rule-based PID controller and a GA optimized fuzzy-PID based neural network controller will be utilized. The AI-based face recognition algorithm will be implemented using Python, and the reference trajectory of the Quadrotor will be determined using the facial positions of the person of interest. The integration of Python-based face recognition will be facilitated through MATLAB® Simulink software.

1.3 Problem Statement

The task at hand is to create a surveillance system that operates independently and utilizes AI-powered face recognition technology. This system will specifically utilize ISR quadrotor UAVs to aid law enforcement agencies in combating terrorism in urban settings, with a particular focus on the activities of terrorist organizations like Alshabab. The conventional control methods do not always provide optimal stability and performance. With the growing complexity and dynamics of modern systems, there is a need for advanced control strategies that can adapt to changing conditions and uncertainties. Therefore, the problem addressed by this research is to explore and develop intelligent control strategies that can enhance stability and performance in ISR Quadrotor UAV.By applying GA Optimized Fuzzy-PID based Neural Network Controller, the control system will enhance the UAV's capabilities to navigate autonomously and make intelligent choices during surveillance missions. The aim is to enhance the efficiency, accuracy, and independence of the UAV in intelligence, surveillance, and reconnaissance operations.

1.4 Objectives

1.4.1 General Objective

The main objective of the research is to model and control ISR Quad Rotor UAV using GA Optimized Fuzzy-PID based Neural Network controller

1.4.2 Specific Objectives

The specific objectives of this research are:

- To develop a mathematical model that describes the behavior and dynamics of the ISR Quadrotor drone.

- To design GA Optimized Fuzzy-PID controller to generate Training Data for NN.
- To design GA Optimized Fuzzy-PID based Neural Network controller for controlling the ISR Quadrotor drone.
- To simulate the overall system using MATLAB®Simulink.
- Design AI based face recognition system using Phyton

1.5 Significance of the Research Work

The research work on "Trajectory Tracking Control of ISR Quadrotor UAV using GA Optimized Fuzzy-PID Based Neural Network Controller"has significant implications in the field of aerial surveillance and intelligence, surveillance, and reconnaissance (ISR) operations. Here are some key points highlighting its significance:

Improved Quadrotor Control: The research aims to enhance the control system of ISR quadrotors by implementing GA Optimized Fuzzy-PID Based Neural Network Controller. This approach can potentially improve the stability, maneuverability, and overall control precision of the quadrotor during ISR operations.

Efficient Surveillance and Reconnaissance: By utilizing a GA Optimized Fuzzy-PID Training Data, the research enhances the performance of the GA Optimized Fuzzy-PID Based Neural Network Controller. This can effectively optimize the ISR quadrotor's ability to perform surveillance and reconnaissance tasks by efficiently navigating through complex environments, maintaining stability, and accurately following predefined trajectories.

Improved Safety and Reliability: The research also focuses on the safety and reliability aspects of the quadrotor. By improving the control system and implementing advanced algorithms, the quadrotor's response to disturbances and changes in parameters be better managed by using the developed controller. This enhances the overall safety and reliability of the ISR quadrotor during missions.

Technological Advancements: The research contributes to the field of robotics and autonomous systems by exploring the integration of neural networks and advanced Training Data from GA Optimized Fuzzy-PID Controller into quadrotor control systems. This has the potential to pave the way for future advancements in aerial robotics, intelligent control systems, and autonomous decision-making algorithms.

Overall, the significance of this research lies in its potential to improve the performance, autonomy, efficiency, and safety of ISR quadrotors, ultimately enhancing the effectiveness of surveillance and reconnaissance operations.

1.6 Methodology

The methodology for Trajectory Tracking Control of ISR Quadrotor UAV using GA Optimized Fuzzy-PID Based Neural Network Controller can be divided into several steps:.

Mathematical Modeling and Model Verification: Based on the Newton Euler formalism, a mathematical model of the quadrotor system is developed. This model should accurately represent the quadrotor's dynamics, kinematics, and control inputs and model verification will be verified check if the modeled Quadrotor represent real Quadrotor.

GA Optimized Fuzzy-PID Controller : The GA Optimized Fuzzy-PID Controller was designed to get input output data for NN. This algorithm combines fuzzy logic, PID control, and genetic algorithms to optimize the performance of the controller. The fuzzy logic component helps in handling uncertain and imprecise inputs, while the genetic algorithm aids in fine-tuning the controller Scaling parameters.

GA Optimized Fuzzy-PID Based Neural Network Controller Design: Next, GA Optimized Fuzzy-PID Based Neural Network Controller is designed. The architecture of the neural network is determined based on the number of inputs and outputs of the control system. The activation functions and network topology can be selected based on the specific requirements of the quadrotor control problem.

Training and Optimization: The GA Optimized Fuzzy-PID Based Neural Network Controller are trained and optimized using appropriate data sets from GA Optimized Fuzzy-PID . This involves feeding the neural network with input-output pairs and adjusting the controller's parameters through the GA Optimized Fuzzy-PID Training Data. The aim is to improve the control response, stability, and robustness of the quadrotor system.

Integration: After the training phase, the GA Optimized Fuzzy-PID Based Neural Network Controller is integrated into the quadrotor system. This may involve developing interfaces to establish communication between the controller and quadrotor model dynamics using MATLAB®simulink.

Performance Evaluation: The performance of the GA Optimized Fuzzy-PID Based Neural Network Controller is evaluated through simulations using MATLAB®Simulink. Various performa nce metrics, such as stability and robustness, are assessed to determine the effectiveness of the proposed control system.

Comparative Analysis: Lastly, a comparative analysis conducted to compare the performance of the GA Optimized Fuzzy-PID Based Neural Network Controller with GA Optimized Fuzzy-PID control methods. This helps to identify the strengths and limitations of the proposed approach in terms of stability and responsiveness to ISR tasks.

By following this methodology, the thesis aims to develop an effective control system for an ISR quadrotor using GA Optimized Fuzzy-PID Training Data, ultimately enhancing its overall performance in ISR operations.

1.7 Scope of the Thesis

The scope of the thesis on Trajectory Tracking Control of ISR Quadrotor UAV using GA Optimized Fuzzy-PID Based Neural Network Controller encompasses several key areas related to the topic:

Modeling the Quadrotor: The thesis will involve developing a mathematical model of the quadrotor system. This includes considering the dynamics, kinematics, and physical characteristics of the quadrotor to accurately represent its behavior. **GA Optimized Fuzzy-PID Controller:** The thesis will explore Data from GA Optimized Fuzzy-PID Controller which will be designed to get input output data for NN.The algorithm combines fuzzy logic, PID control, and genetic algorithms to optimize the controller's performance. **GA Optimized Fuzzy-PID Based Neural Network Controller Design:**The thesis will focus on designing a GA Optimized Fuzzy-PID Based Neural Network Controller for the quadrotor. This involves developing a suitable neural network architecture and training

it using appropriate algorithms to enable effective control of the quadrotor's flight dynamics. **Integration:** The thesis will involve integrating the GA Optimized Fuzzy-PID Based Neural Network Controller and GA Optimized Fuzzy-PID into the quadrotor system. This includes using MATLAB®simulink software to interfaces to enable communication between the controller and the quadrotor's Dynamics. **Performance Evaluation:**The thesis will evaluate the performance of the quadrotor system with the implemented GA Optimized Fuzzy-PID Based Neural Network Controller and GA Optimized Fuzzy-PID Controller. This includes conducting simulations to assess the system's stability and maneuverability in ISR operations.**Comparative Analysis:**The thesis may also include a comparative analysis of the a GA Optimized Fuzzy-PID Based Neural Network Controller with the GA Optimized Fuzzy-PID control methodologies for quadrotors. This allows for insights into GA Optimized Fuzzy-PID Based Neural Network Controller using the GA Optimized Fuzzy-PID Training Data in terms of performance, robustness, and stability.

The scope of the thesis primarily revolves around developing an improved control system for ISR quadrotors which is GA Optimized Fuzzy-PID Based Neural Network Controller by using GA Optimized Fuzzy-PID Training Data . The focus is on enhancing the quadrotor's stability, maneuverability, and overall control performance to improve its effectiveness in ISR tasks.

1.8 Outlines of the Thesis

The thesis is arranged into seven chapters it were arranged here as order. **Chapter 2**: Describes the work of others witch is review of Literature's(past work done by others)about Overview of ISR Quadrotor UAV and its importance in ISR Applications,Review of Face recognition and Identification using Phyton and Review of Controllers to Control Unstable Dynamics of Quadrotor UAV furthers Discussed. **Chapter 3**: In focus of modeling physical system behaviour into differential equations of the Quad Rotor UAV including Verification of Modeled Quadrotor at different Flight condition was analyzed. **Chapter 4**: Design GA Optimized Fuzzy-PID Controller and GA Optimized Fuzzy-PID Based Neural Network Controller ,the controllers are designed in this chapter and the proposed controller for each dynamics was designed and the trained result was discussed. **Chapter 5**: Focuses on the Face identifications using Phyton & Furthermore the way of Implementation are Discussed including Autonomous tracking and semi autonomy operational scenario was discussed in this chapter . **Chapter 6**:Simulation Results and Discussion ,the Plant response for both controllers in different trajectories in addition with Step Response and the performance of the controllers with input Disturbance and Parameter Variation was discussed and the robustness of Controllers checked based on Performance metrics. **Chapter 7**: Draws Conclusion from the work done(Simulation Results) and Recommend Recommendation for Future Works.

Chapter 2

Literature Review

This chapter presents a comprehensive review of relevant research articles and studies that focus on the control aspects of Quad-rotors UAV. Various controllers, ranging from classical control to nonlinear, optimal control, and intelligent control techniques, are explored. Addition ally, the chapter discusses the use of Python for face recognition and identification.

2.1 Overview of ISR Quadrotor and It's Importance in ISR Application

ISR (Intelligence, Surveillance, and Reconnaissance) quadrotors play a significant role in various ISR applications. These unmanned aerial vehicles (UAVs) are equipped with advanced technological capabilities that enable them to gather intelligence, monitor activities, and provide reconnaissance in both military and civilian contexts [1].

Here is an overview of ISR quadrotors and their importance in ISR applications:

Aerial Surveillance: ISR quadrotors enable aerial surveillance over a specific area or target. Equipped with high-resolution cameras, thermal imaging systems, and sensors, they can capture real-time imagery and gather valuable intelligence from the air. This capability is crucial for military operations, border control, law enforcement, and disaster response scenarios.

Reconnaissance: Quadrotors equipped with advanced sensors and imaging systems can conduct surveillance and reconnaissance missions in various environments. They can capture crucial information, identify targets or threats, and provide in-depth situational awareness to ground commanders or operators. This helps in decision-making and planning operations effectively.

Intelligence Gathering: ISR quadrotors are designed to collect intelligence data from remote or inaccessible regions. Their maneuverability and ability to navigate challenging terrains allow them to reach areas that are otherwise difficult to access. By gathering data such as imagery, signals, or electronic intelligence, they contribute to timely and accurate intelligence analysis.

Target Tracking: Quadrotors with sophisticated tracking systems can monitor and track specific targets or individuals. This is particularly useful in law enforcement operations, border surveillance, and tracking potential threats. The UAV's agility and ability to maintain visual contact make them ideal for tracking subjects both in urban and rural areas.

Rapid Response and Situational Awareness: ISR quadrotors provide rapid response capabilities, enabling quick deployment for emergency situations. They can provide real-time situational awareness in disaster zones, search and rescue missions, or post-conflict areas. Their aerial perspective assists in locating survivors, assessing damage, and coordinating relief efforts.

Force Multiplier and Risk Mitigation: By deploying ISR quadrotors, operational forces can extend their reach and enhance their capabilities. Quadrotors act as force multipliers by augmenting human presence, reducing risks, and providing crucial information without endangering personnel. They offer increased flexibility, persistence, and coverage in ISR operations.

ISR quadrotors have proven their significance in a wide range of applications, including military intelligence, law enforcement, border security, disaster management, and environment al monitoring. Their versatility, maneuverability, and advanced technological features make them valuable assets for gathering intelligence, conducting reconnaissance, and maintaining situational awareness. By leveraging these capabilities, operators can make informed decisions, enhance operational effectiveness, and ensure the safety of their personnel.

Drone Intelligent surveillance& Reconnaissance provides real-time insight into security and emergency situations for better control, accurate intelligence gathering,comprehensive situational awareness, and more informed decision making [2] [3].Experts in intelligence, surveillance, and reconnaissance (ISR) observe the enemy's conduct and follow their movemen ts to learn more about them at the situation room or surveillance area. Mission success is greatly enhanced by staying three steps ahead of the opposition. Battlefield Military commanders use ISR capabilities to gather relevant information for combat planning at battlefield and the Urban based Law enforcers use ISR capabilities for tracking terrorists and its recruitment cells in the city. This allows them to intercept communications, monitor movements and develop plans, strategies, and allocate resources for operational mission accomplishments.

Accurate ISR data are critical for furnishing high-quality intelligence about adversary pitfalls and serve to increase the effectiveness of military operations effectively and efficiently at no or low cost of human power. The ongoing development of technology has only increased the demand for ISR capabilities in recent times for Protecting the National security of Nations and it's interests.

2.2 Face Recognition and Identification

Computerized facial recognition is a relatively new technology, being introduced by law enforcement agencies around the world in order to identify persons of interest to obey rule of law [4]. according to Kelley M Sayler(2020) [5]In 2021, the Portland City Council implemented one of the most stringent regulations in the United States concerning the use of facial recognition technology with drones. The New York Civil Liberties Union emphasized the concerns surrounding drones, which have the potential to be equipped with diverse surveillance features such as facial recognition, emotion recognition, and behavior detection. The organization underscored that these systems can be unreliable and give rise to erroneous arrests.

Furthermore, it has come to light that the U.S. Air Force has developed the capability

to employ facial recognition on drones, enabling them to target individuals with precision. which is wanted for their crime committed Kelley M Sayler(2020) [5].

Vera L´ucia Raposo(2022) [6]suggests that crime investigation using facial recognition technology to enforce rule of law,One of the main benefits of facial recognition technology is its ability to quickly and accurately identify individuals(Person of interest). This can be particularly useful in criminal investigations(wanted persons identification and recognition), where facial recognition can be used to identify suspects and match them to surveillance footage using this technology. Additionally, facial recognition can be used to track individuals who may be wanted by law enforcement, such as fugitives or known wanted personals.This facial recognition technology by law enforcement also raises a number of concerns according to Vera L´ucia Raposo(2022) [6]. One of the main concerns is the potential for misuse and abuse of the technology(using the technology for themselves rather than using it for interest of nations).

2.3 Control of Quadrotor

To do the above mission the followings are the Review of Literature for controllers to control unstable Dynamics of Quadrotor,hence the next are the previous works done by others regarding controlling non linear quad-rotor. Quadrotors, also known as quadcopters, are multirotor helicopters powered by four rotors. They are highly maneuverable and find numerous applications in areas such as aerial surveillance, transportation, search and rescue, and entertainment. However, quadrotors exhibit inherently unstable dynamics, making it challenging to control their movements accurately and efficiently. This review aims to explore the existing literature on different control algorithms applied to quadrotors to stabilize their flight and enhance their performance.

Rafael Siegwart and Illah R. Nourbakhsh (2013) [7]This book chapter presents an in-depth discussion of quadrotor control. It covers different control architectures, state estimation techniques, feedback control approaches, and guidance strategies. The Journal Paper PID control of quadrotor for an autonomous landing on a moving platform by Yongquan Chen and Xiangyu Chen (2017) [8] this works focuses on the PID control of a quadrotor for autonomous landing on a moving platform. It presents a mathematical model of the quadrotor system and designs a PID control scheme to stabilize the vehicle during landing.

Dong et al.(2018) [9] presented a PID controller for a quadrotor, incorporating an extended state observer to estimate disturbances and a proportional-integral (PI) controller to suppress disturbances effectively. The controller demonstrated efficient disturbance rejection and improved trajectory tracking the limitation is the controller was Linear.

Li et al. (2016) [10] proposed an MPC-based control algorithm for quadrotors, enabling real-time estimation of the quadrotor's states and control inputs. The MPC controller achieved precise trajectory tracking and robustness against external disturbances. Chang et al. (2014) [11] introduced an adaptive sliding mode control approach for quadrotors. The controller utilized a novel adaptive law to estimate uncertain parameters. The adaptive control scheme proved effective in handling quadrotor dynamics under uncertain conditions and disturbances. Ke et al. (2017) [12] developed a linear quadratic regulator (LQR) controller for quadrotors. The LQR controller exploited an optimal control formulation to minimize a cost function and stabilize the quadrotor's dynamics. The controller demonstrated

superior stability and control performance. Feng et al.(2016) [13]proposed a nonlinear control algorithm based on the backstepping method to stabilize quadrotor dynamics. The controller accounted for nonlinearities in the system dynamics and achieved robust stability and accurate trajectory tracking. Zhu et al(2022) [14]introduced a neural network-based model-free adaptive control approach for quadrotors. The controller utilized a radial basis function neural network to approximate the unknown quadrotor dynamics. The neural network controller demonstrated excellent trajectory tracking and disturbance rejection capabilities.

Ahmed Eltayeb, Mohd Fuaad Rahmat (2021) [15] Proposed that Fuzzy PID Control Approach for Quadrotor trajectory Tracking and the fuzzy based self tuning Controller improves performance of tracking by 17% than that of conventional PID controller and the self tuning PID using fuzzy set was adapted for external input disturbance but the limitation in this work is that the simulation time is too short. El Hamidi(2019) [16] Proposed that Neural network and fuzzy-logic-based self-tuning PID control for quad-copter path tracking ,in this work the adaptive PID parameters are tuned using fuzzy sets and Neural Network system and the scaling factors are optimized using Particle Swarm Optimization(PSO) Algorithm hence according to El Hamidi(2019) [16]the adaptive,optimized and Intelligent controller has high performance than conventional controller by rejecting the input disturbance.

Mebaye Belete Mam(2020) [17] suggests that third order super twisting Sliding Mode Control(SMC) to control the nonlinear dynamic of Quadrotor in this work altitude and attitude dynamics controller was good tracking Performance but the limitation or drawback is that the chattering is not reduced. L Salih(2010) [18]suggests that PID control of Quadrotor dynamics the performance of controller was good but the plant dynamics was linearized and it was hard to control linear plant at the moment when occurrences of external disturbance like wind forces.

Yang Chen(2018) [19]proposed LQR control approach for quad-rotor trajectory tracking and again in this work the plant dynamics was linearized and even if the controller performance was so good it doesn't fit in real world operational scenario,therefore its hard to control during implementation in real consideration and again the controller itself was linear controller and doesn't handle highly nonlinear Plant like ISR Quadrotor.

Lee(2013) [20]The proposed approach in this work involves utilizing Feedback Linearization with LQR control to achieve quad-rotor trajectory tracking. The controller's performance has been found to be exceptionally good for both rectangular and infinity trajectory tracking.accordingly to the Author and the non Linear Mathematical model was derived based on Newton Euler formalism and the non linear controller which is feedback Linearization controller was designed to control the Plant but the drawback was using classical nonlinear controller was not as better as modern Intelligent controllers like Fuzzy and Neural Network controllers. Tolga (2017) [21]proposed Neural Network control of Quadrotor based on PID learning algorithm the simulation result track the reference trajectory in better way but the limitation in this work was that the learning algorithm controller which means the training data is from PID which is classical linear controller ,therefore it's hard to control the plant in the case of wind which is external disturbance and when the plant parameter is also varied the controller performance is not good.

This literature review highlights various control algorithms applied to handle the unstable dynamics of quadrotors. The reviewed controllers, including PID, MPC, adaptive control, optimal control, nonlinear control, and neural network control, offer effective solutions for

stabilizing quadrotor flights, achieving accurate trajectory tracking, and enhancing disturbance rejection capabilities. These findings can guide researchers and engineers in selecting appropriat e control strategies for controlling the unstable dynamics of quadrotors. Further research can focus on hybrid control schemes, robust control techniques, and advanced machine learning-based algorithms to improve the control performance of quadrotors.

Hence considering the works of others(Literature Review) most researchers use the linear plant model even if using the intelligent controllers like Neural Network control so that this thesis work uses the nonlinear plant model and the nonlinear control techniques both for training data and proposed controllers which is Fussy rule based tuned PID controllers with scaling factors are again tuned automatically using Genetic Algorithm Optimization Algorithm and the GA Optimized Fuzzy-PID Based Neural Network Controller was Proposed.

In this thesis work external disturbances and parameter Variation Applied at the same time at controllers that are omitted in many literature's are given due attention Moreover,using intelligent control Techniques GA Optimized Fuzzy-PID Controller and GA Optimized Fuzzy-PID Based Neural Network Controller was proposed used to improve robustness and trajectory tracking performances of ISR Quadrotor UAV.

Chapter 3

Modelling of Quad-Rotor UAV

3.1 Modeling Approach

In this Chapter the Quad-rotor geometrical structure and its reference frame are discussed and also its nonlinear dynamic model will be developed based on Euler Newton formalism.

3.1.1 Types of Configuration of Quad-Rotor

There are many configurations for multi rotor drones but for quad-rotors have has two known configurations those are plus(+)configuration and cross (x)configurations .from the two configurations this Thesis work mainly focuses on cross(x)configuration depending on the mission that the drone is designed for and the mission is ISR to do it the cross(x)configuration is recommended in this Thesis.

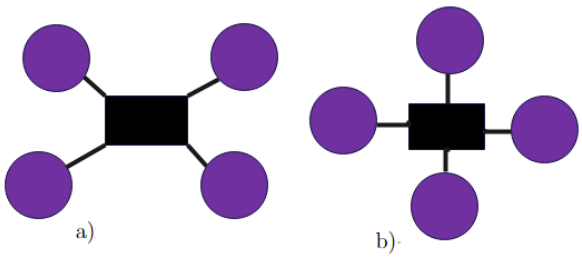


Figure 3.1: a)cross (x)configuration,b)plus(+)configuration

In this Thesis the Quad rotor is used to deploy for law enforcement mission hence,the cross configuration is used depending on mission specification the drone is used for and below are the advantages of cross configuration over plus configuration for Quad-rotor.

- **The drone is used for law enforcement mission** and can carry intelligent payload(ca mera) hence,the cross configuration will be selected for this mission and where as plus one can hide the payload with its frame,hence it's omitted.

3.2 Reference Frame Alignment

A reference frame is a set of points in space for which the distance between any two points is fixed at all times [22]. Therefore,any choice of the three or more non-collinear points can

be used as a frame of reference called reference frame.hence, The simplest way to study a reference frame is to think of a rigid body system. Rigid bodies and reference frames are thus used interchangeably. Therefore, a two-coordinate systems was required for. which are the body and inertia or earth frame of referances,There is the body frame system which is attached to the ISR Quad-Rotor at its center of gravity and the earth frame system which is fixed to the earth and it is sometimes refer to as an inertial coordinate system from the vehicle manipulator book as defined in clarity and concise way.

The BFF is attached to the quad-rotor itself and is centered around its center of gravity. This frame allows for readings from sensors like gyroscope and accelerometers, providing information relevant to the quad-rotor's movement and orientation.

On the other hand, the Earth (inertial) reference frame is fixed to the Earth and typically represented by North-East-Up coordinates. This frame is used for sensors like GPS and magnetometer, which give readings with respect to the Earth's reference frame.

By having these two frames, the quad-rotor can gather data from different sensors and apply appropriate transformations to derive system equations in a consistent frame of reference. To align the reference frame of an ISR Quadrotor, the goal is to establish a consistent and standardized coordinate system that can be used for measurements and control. Here are the steps involved in the process:

Body Frame Alignment:

- Mount the quadrotor on a level surface.
- Ensure that the center of gravity of the quadrotor is at the center of the frame.
- Establish the x, y, and z axes of the body frame. The most common convention is to align the x-axis with the forward direction, the y-axis with the upward direction, and the z-axis downward

Inertial Frame Alignment:

- The inertial frame is typically represented by a North-East-Up (NEU) coordinate system.
- Calibrate the quadrotor's onboard sensors such as GPS and magnetometer to obtain accurate readings.
- Perform sensor fusion algorithms to align the inertial frame with the Earth's reference frame.
- To establish the origin of the inertial frame, select a fixed point on the surface of the Earth.

Transformations:

- To work with a single reference frame, transformation matrices or equations can be used to convert measurements between the body frame and the inertial frame.
- For example, the readings from the gyroscopes and accelerometers in the body frame can be transformed to the inertial frame using appropriate rotation matrices.

By aligning the reference frames of the ISR Quadrotor, it becomes easier to interpret and analyze sensor data, perform control calculations, and navigate the quad-rotor accurately. And to analyse the dynamics motion of the ISR Quadrotor UAV, the two frames of references were specified as below:

3.2.1 Body-fixed Reference Frame(BFF)

BFF(o^B, x^B, y^B, z^B)

- 0^B coincides with center of the quad-rotor
- Linear Velocity$(V^B = \begin{bmatrix} U, & V, & W \end{bmatrix}^T$
- Angular Velocity $(\omega^B = \begin{bmatrix} p, & q, & r \end{bmatrix}^T$
- Force(F^B)
- Torque(τ^B)

3.2.2 Earth or Inertial Reference Frame(IFF)

IFF $(o^E$,x^E,y^E,$z^E)$

To specify the linear position$(L^E = \begin{bmatrix} x, & y, & z \end{bmatrix}^T)$and the angular location(position)$(A^E = \begin{bmatrix} \phi, & \theta, & \psi \end{bmatrix}^T)$

L^E The coordinates of the vector that connectsO^B and O^E with regard to the IFF & A^E use the right hand rule to depict the reference frames and the orientation of the BFF with regard to the IFF.

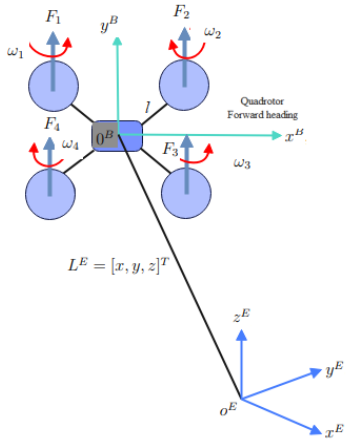


Figure 3.2: Reference Frame

3.2.3 Assumptions for Quad-Rotor Modelling and its Operation Principles

The assumptions mentioned are commonly made when deriving the nonlinear dynamic model of a quadrotor UAV [23].

- **Rigid and symmetrical structure:** This assumption implies that the quadrotor has a fixed shape and its mass is uniformly distributed. It simplifies the modeling process by considering the quadrotor as a rigid body.
- **Center of gravity:** Assuming that the quadrotor's center of gravity coincides with the body-fixed frame origin simplifies the analysis of the system dynamics. This assumption eliminates the need to consider the translation of the center of gravity.
- **Rigid propellers:** Assuming that the propellers are rigid simplifies the modeling of the forces and moments generated by the propellers. It implies that the blades of the propellers do not flex or deform during operation.
- **Thrust and drag force:** The assumption that the thrust and drag forces are proportional to the square of the propeller's speed is based on empirical data and observations. In practice, the relationship between thrust/drag and speed can be approximated by quadratic functions.
- **Motor dynamics:** Neglecting the motor's resistance and inductance implies that the motor dynamics are significantly faster than the dynamics of the quadrotor itself. This assumption simplifies the modeling process by assuming that the motor instantaneously responds to changes in voltage and produces the corresponding output speed.

  It's important to note that these assumptions introduce simplifications and idealizations, allowing for a simplified mathematical description of the quadrotor dynamics. However, in practice, some of these assumptions may not hold perfectly, and more complex models may be required for more accurate analyses or control designs.

3.2.3.1 Operation Principle of Quad-Rotor

The operation principles of an ISR quadrotor (Intelligence, Surveillance, and Reconnaissance) are based on the control of the four propellers. Here's a breakdown of the key principles [24]:

- **Configuration:** A quadrotor consists of four propellers positioned at the four corners of the frame. Each propeller generates lift and thrust for the vehicle.
- **Balance and Stability:** To maintain balance and stability, the quadrotor operates on the principle of counteracting torques. One pair of propellers rotates in a clockwise direction, while the other pair rotates in an anticlockwise direction. The opposite rotations create counteracting torques, which help keep the quadrotor stable and level.
- **Takeoff:** To take off, all four propellers accelerate and run at a high speed, usually at full throttle. The increased speed generates enough lift to overcome the gravitational force, allowing the quadrotor to lift off the ground.
- **Pitch, Roll, and Yaw:** To maneuver in different directions, the quadrotor adjusts the speed of individual propellers. By increasing or decreasing the speed of the propellers on different sides of the quadrotor, it can tilt and move in different directions.

  Pitch: Tilting the quadrotor forward or backward is achieved by increasing or decreasing the speed of the front or rear propellers respectively.

Roll: Tilting the quadrotor sideways is accomplished by increasing or decreasing the speed of the left or right propellers respectively.

Yaw: Changing the orientation of the quadrotor around its vertical axis (yaw) is achieved by increasing or decreasing the speed of the propellers on one side while simultaneously decreasing or increasing the speed of the propellers on the other side.

By carefully adjusting the speed of the propellers, the quadrotor can move in any desired direction.

It's important to note that the precise control of the quadrotor's movements is achieved through sophisticated flight control algorithms and sensor integration, such as gyroscopes, accelerometers, and GPS, which provide feedback to the control system. These control systems continuously adjust the propeller speeds to maintain a stable and controlled flight

3.2.4 Quad-Rotor Dynamics

Quad-rotor dynamics refers to the movement and behavior of a drone [25], which can be categorized into four types based on the motion of the four propellers: throttle, pitch, roll, and yaw. Let's take a closer look at each of these types:

- **Throttle/Hover:** Throttle refers to the upward and downward movement of the quad-rotor. When all four propellers operate at a normal speed, the drone will move downwards. Conversely, when all four propellers run at a higher speed, the drone will move upwards. This is known as the hovering condition of a drone, indicating the amount of thrust applied to the quad-rotor's body.

- **Pitch:** Pitching motion refers to the forward or backward movement of the drone along the lateral axis. When the two rear propellers run at high speed, the drone will move in a forward direction. Conversely, when the two front propellers run at high speed, the drone will move backward. The verification of the pitch model is performed after completing the dynamic model.

- **Roll:** Rolling motion refers to the movement of the drone along the longitudinal axis. When the two right propellers run at high speed, the drone will move to the left. Similarly, when the two left propellers run at high speed, the quad-rotor will move to the right. The verification of the roll model is carried out after completing the dynamic model.

- **Yaw:** Yawing motion refers to the rotation of the quad-rotor's head around the vertical axis, either to the left or right. When the two propellers of the right diagonal run at high speed, the drone will rotate in an anti-clockwise direction. Conversely, when the two propellers of the left diagonal run at high speed, the drone will rotate in a clockwise direction. The verification of the yaw model is performed after completing the dynamic model.

  These dynamics play a crucial role in understanding and controlling the movements and behavior of quad-rotors.

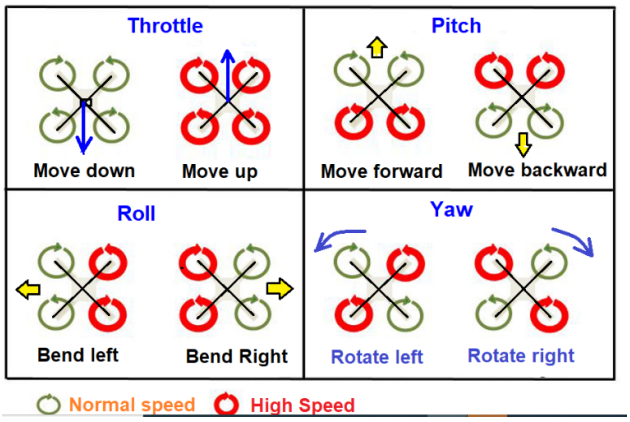


Figure 3.3: Working Principle of Quad-rotor [26]

3.2.4.1 Basic Forces Applied to Quad Rotor Body

The basic forces applied to an ISR Quadrotor body are lift, drag, weight, and thrust. Here's a breakdown of each force:

- **Thrust force:** The thrust force is responsible for propelling the Quadrotor through the air in an upward direction, opposing the force of drag. Thrust is used to overcome the drag of the drone, thus enabling it to move and maintain flight.
- **Drag force:** Drag is a force that operates in opposition to the quadrotor's relative motion. as it moves through a fluid, which can be air in this case. It can exist between layers of fluid or between a fluid and a solid surface. Drag opposes the lift force of the Quadrotor.
- **Weight:**Weight is the force exerted by the Earth on the Quadrotor due to gravity. The Earth pulls all objects, including the drone, downward towards its center. The magnitude of the weight force can be calculated by multiplying the mass (m) of the Quadrotor by the acceleration due to gravity (g). These forces work together to enable the Quadrotor to fly and maneuver in the air.

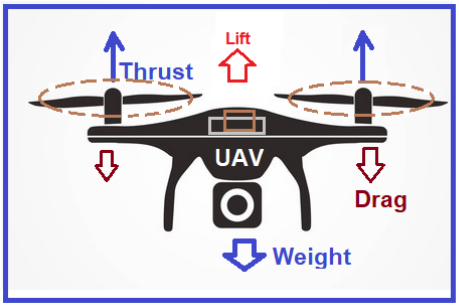


Figure 3.4: Basic Forces Applied to Quad-rotor [26]

3.2.5 Kinematics for Quad-Rotor

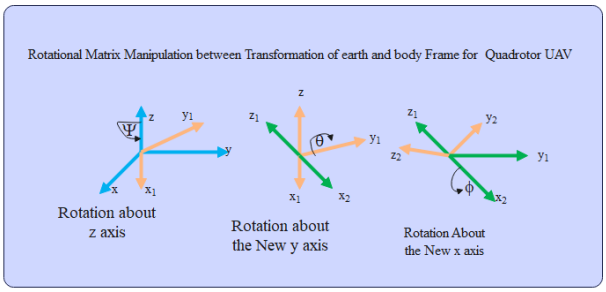
Kinematics is a branch of mechanics that focuses on studying the motion of a system or plant (such as a quadrotor) Without thinking the forces and torques acting upon it. In

order to describe the motion of a rigid body with 6DOF it is beneficial to establish two RF, as depicted in Figure (3.2) in the previous section.

Rotation Matrix

Quad-rotor employs different control mechanism such as roll, pitch, and yaw which in most cases are represented by Euler angle of rotation around the center of the quad-rotor. To bring the body fixed frame into coincidence with the earth fixed frame the rotation matrix are considered to used. It describes the transformation earth-fixed coordinate to body fixed coordinates by using rotational matrix [22]

The rotation system can be derived by rotating the body frame around the z axis of the Quad-rotor by yaw angle ψ, followed by rotating around the y axis by the pitch angle θ and finally by rotating around the x-axis by the roll angle ϕ with respect to fixed reference frame. In order to avoid the system singularities (loss of degree of freedom in space) it is sufficient to assume the pitch angle and roll angle between -90 and 90 degree and the Yaw angle between -180 and 180 degree.



$$R(x,\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\phi & sin\phi \\ 0 & -sin\phi & cos\phi \end{bmatrix}, R(y,\theta) = \begin{bmatrix} cos\theta & 0 & -sin\theta \\ 0 & 1 & 0 \\ sin\theta & 0 & cos\theta \end{bmatrix}, R(z,\psi) = \begin{bmatrix} cos\psi & sin\psi & 0 \\ -sin\psi & cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

The orientation of the quad-rotor is given by the rotation matrix R which depends on the three Euler angles (ϕ,θ,ψ) and defined by the following equation after combined.

$$\begin{aligned} {}^{B}_{E}R &= R_{xyz} = R(z,\psi)R(y,\theta)R(x,\phi) \\ &= \begin{bmatrix} cos\phi cos\theta & cos\theta sin\psi & -sin\theta \\ sin\phi sin\theta cos\psi - cos\phi sin\psi & sin\phi sin\theta sin\psi + cos\phi cos\psi & sin\phi cos\theta \\ cos\phi sin\theta cos\psi + sin\phi sin\psi & cos\phi sin\theta sin\psi - sin\phi cos\psi & cos\phi cos\theta \end{bmatrix} \end{aligned} \tag{3.2}$$

$${}^{E}_{B}R = \mathrm{R}^{\mathrm{T}}_{\mathrm{xyz}} = \begin{bmatrix} \mathrm{cos}\phi\mathrm{cos}_{\theta} & \mathrm{sin}\phi\mathrm{sin}\theta\mathrm{cos}\Psi - \mathrm{cos}\phi\mathrm{sin}\Psi & \mathrm{cos}\phi\mathrm{sin}\theta\mathrm{cos}\Psi + \mathrm{sin}\phi\mathrm{sin}\psi \\ \mathrm{cos}\theta\mathrm{sin}\psi & \mathrm{sin}\phi\mathrm{sin}\theta\mathrm{sin}\psi + \mathrm{cos}\phi\mathrm{cos}\Psi & \mathrm{cos}\phi\mathrm{sin}\theta\mathrm{sin}\psi - \mathrm{sin}\phi\mathrm{cos}\Psi \\ -\mathrm{sin}\theta & \mathrm{sin}\phi\mathrm{cos}\theta & \mathrm{cos}\phi\mathrm{cos}\theta \end{bmatrix} \tag{3.3}$$

$$\begin{bmatrix} \frac{dx}{dt} \\ \frac{dy}{dt} \\ \frac{dz}{dt} \end{bmatrix} = {}^{E}_{B}R \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{3.4}$$

Transfer Matrix $A^E = [\phi, \theta, \psi]^T$ is in IFF and Angular Velocity $\left(\omega^B = [p, q, r]^T\right)$ is in BFF to relate using a transfer matrix(T): $\dot{A}^E = T\omega^B$

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = T \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$
$$T = \begin{bmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi)\sec(\theta) & \cos(\phi)\sec(\theta) \end{bmatrix}, T^{-1} = \begin{bmatrix} 1 & 0 & -\sin(\theta) \\ 0 & \cos(\phi) & \sin(\phi)\cos(\theta) \\ 0 & -\sin(\phi) & \cos(\phi)\cos(\theta) \end{bmatrix} \tag{3.5}$$

The detail manipulation of Transfer matrix was at Appendix A.

Newton-Euler Method formalism The flight dynamics of the ISR QuadRotor will be modeled using the Newton-Euler method. The equations of motion, which combine the translational and rotational parameters of the 6DOF system (consisting of 3 position and 3 angular orientation variables) [27], are derived based on Euler's two laws of motion – Newton's second law and Newton's first law (also known as the law of inertia). The formulation commonly involves the following parameters: - Linear Velocity $\left(V^B = [u, v, w]^T\right)$ - Angular Velocity $\left(\omega^B = [p, q, r]^T\right)$ - Orientation Vector $\left(A^E = [\phi, \theta, \psi]^T\right)$ - Position Vector $\left(L^E = [x, y, z]^T\right)$

$$\begin{bmatrix} \text{Roll rate} \\ \text{Pitch rate} \\ \text{Yaw rate} \end{bmatrix} = \begin{bmatrix} p \\ q \\ r \end{bmatrix}_B \tag{3.6}$$

External Forces and Moments

Force: The force exerted on an ISR quad-copter affects its translational motion. According to Newton's second law, this force can be expressed as : $\sum F = F_{\text{th}} + F_{\text{grav}} = m\vec{a}$. In the body-fixed reference frame, the translational dynamics are influenced by two main forces.

Thrust Force: Firstly, the thrust force is the combined thrust generated by the propellers to lift the ISR quad-copter upwards. The thrust is related to the angular velocity through a quadratic relationship $(F_i = b\omega_i^2)$ where: b represents a constant that depends on various factors like the air density, propeller area, back electromotive force, and torque proportionality constant [27] [28].

$$F_{\text{th}}^B = F_1 + F_2 + F_3 + F_4 = b\omega_1^2 + b\omega_2^2 + b\omega_3^2 + b\omega_4^2 \tag{3.7}$$

Gravity of Attraction: Second, the force of gravity acting on the quad-rotor UAV is the gravitational force, which only acts in the Z-direction. Although Euler angles can also be used to translate it with respect to the body-fixed frame, the Earth inertial frame provides

a more accurate expression.

$$F^B_{\text{grav}} = \left({}^E_B R\right)^{-1} F^E_{\text{grav}}\text{, where: } \left({}^E_B R\right)^{-1} = \left({}^E_B R\right)^T\text{, and } F^E_{\text{grav}} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} \tag{3.8}$$

$$F^B_{grav} = \begin{bmatrix} mg\sin(\theta) \\ -mg\cos(\theta)\sin(\phi) \\ -mg\cos(\theta)\cos(\phi) \end{bmatrix} \tag{3.9}$$

Using Newton Euler method, Euler's 1st axioms of Newton 2nd law: $F = m\vec{a}$, in body-fixed frame $\sum F^B = m\frac{d\vec{v}}{dt} \rightarrow \vec{v} = ui + vj + wk$, The acceleration is found by deriving, $\vec{v}$. $\frac{d\vec{v}}{dt} = \frac{du}{dt}i + \frac{dv}{dt}j + \frac{dw}{dt}k + \frac{di}{dt}u + \frac{dj}{dt}v + \frac{dk}{dt}w$ By the way The derivative would simply be the derivative of the velocity magnitude in each of the .i, j, k components of the reference frame if body orientation was constant. Hence, the unit vectors (i, j, k) would be constant, meaning that their derivative is null. The chain rule derivation should be taken into consideration when dealing with rotating bodies. Following the cross-multiplication:

$$\sum F^B = \begin{bmatrix} F_x \\ F_y \\ F'_z \end{bmatrix} = \begin{bmatrix} m(\dot{u} + (qw - vr)) \\ m(\dot{v} + (ru - pw)) \\ m(\dot{w} + (pv - uq)) \end{bmatrix} \tag{3.10}$$

$$\sum F^B = F^B_{\text{th}} + F^B_{\text{grav}} = m\vec{a} \tag{3.11}$$

$$\underbrace{\begin{bmatrix} 0 \\ 0 \\ F_1 + F_2 + F_3 + F_4 \end{bmatrix}}_{F^B_{\text{th}}} + \underbrace{\begin{bmatrix} mg\sin(\theta) \\ -mg\cos(\theta)\sin(\phi) \\ -mg\cos(\theta)\cos(\phi) \end{bmatrix}}_{F^B_{\text{grav}}} = \underbrace{\begin{bmatrix} m(\dot{u} + (qw - vr)) \\ m(\dot{v} + (ru - pw)) \\ m(\dot{w} + (pv - u)) \end{bmatrix}}_{Ma} \tag{3.12}$$

$$\begin{aligned} \dot{u} &= g\sin(\theta) - (qw - vr) \\ \dot{v} &= -g\sin(\phi)\cos(\theta) - (ru - pw) \\ \dot{w} &= -g\cos(\phi)\cos(\theta) - (p - uq) + \frac{F^B_{\text{th}}}{m} \end{aligned} \tag{3.13}$$

The translational forces are described using the Earth's inertial reference frame. The rotation matrix can be used to obtain the earth inertia frame description of the trust force, which is described in a BFF.

$$\left({}^E_B R\right) F^E_{\text{th}} = {}^E_B R * F^B_{\text{th}} \tag{3.14}$$

$$F^E_{\text{th}} = \begin{bmatrix} (\cos(\phi)\sin(\theta)\cos(\psi) + \sin(\phi)\sin(\psi))F^B_{\text{th}} \\ (\cos(\phi)\sin(\theta)\sin(\psi) - \sin(\phi)\cos(\psi))F^B_{\text{th}} \\ (\cos(\phi)\cos(\theta))F^B_{\text{th}} \end{bmatrix} \tag{3.15}$$

Gravitational Force: is expressed in the earth inertial frame described as below form.

$$F_{grav}^{E} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} \tag{3.16}$$

Newton's second law's first axioms, as established by the Newton Euler method: In the earth's inertial reference frame,: $F = M\vec{a}$, in earth inertial frame of reference were depicted as:

$$\sum F^{E} = F_{\text{th}}^{E} + F_{graw}^{E} = m\vec{a} \tag{3.17}$$

$$\underbrace{\begin{bmatrix} (\cos(\phi)\sin(\theta)\cos(\psi) + \sin(\phi)\sin(\psi))F_{th}^{B} \\ (\cos(\phi)\sin(\theta)\sin(\psi) - \sin(\phi)\cos(\psi))F_{\text{th}}^{B} \\ (\cos(\phi)\cos(\theta))F_{\text{th}}^{B} \end{bmatrix}}_{F_{\text{th}}^{E}} + \underbrace{\begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix}}_{F_{\text{grav}}^{E}} = m \underbrace{\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix}}_{m\vec{a}} \tag{3.18}$$

Let:

$$U_1 = F_{\text{th}}^{B} = b\left(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2\right) \tag{3.19}$$

The equation defining the dynamics of transitional motion in the earth's inertial reference frame were

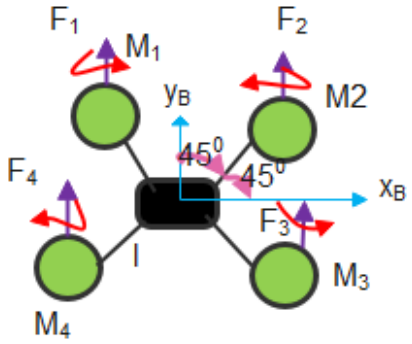
$$\begin{aligned} \ddot{x} &= (\cos(\phi)\sin(\theta)\cos(\psi) + \sin(\phi)\sin(\psi))\frac{U_1}{m} \\ \ddot{y} &= (\cos(\phi)\sin(\theta)\sin(\psi) - \sin(\phi)\cos(\psi))\frac{U_1}{m} \\ \ddot{z} &= (\cos(\phi)\cos(\theta))\frac{U_1}{m} - g \end{aligned} \tag{3.20}$$

Torque: A quadcopter's rotational motion is influenced by torque. By taking into account the developed net torque and reaction forces, the rotational dynamics can be formulated similarly to the developed translational dynamics: $\sum \tau = M_{\text{thrust}} + F_{gyro}$. It is possible to determine the quadrotor's rotational dynamics by utilizing both frames.

In Body-fixed reference frame: Moments of thrust forces: $\sum (r_i \times F_i)$, is the moment of thrust forces in the body-fixed reference frame. The reaction moments have a quadratic relationship with the angular velocity. $M_i = l\omega_i^2$, constant $l.l$ depends on parameters such as material, air viscosity, pitch, diameter, and number of blades on the propellers. The sum of the moments in each coordinate is equal to:

$$\begin{aligned} \tau_x^B &= -F_1 l\cos(45^\circ) + F_2 l\cos(45^\circ) + F_3 l\cos(45^\circ) - F_4\cos(45^\circ) \\ \tau_y^B &= F_1 l\sin(45^\circ) + F_2 l\sin(45^\circ) - F_3 l\sin(45^\circ) - F_4 l\sin(45^\circ) \\ \tau_z^B &= -M_1 + M_2 - M_3 + M_4 \end{aligned} \tag{3.21}$$

where, l is the length between the center of the ISR Quadrotor propeller and the quadrotor itself.



$$\begin{aligned}
\tau_x^B &= -F_1 l\frac{\sqrt{2}}{2} + F_2 l\frac{\sqrt{2}}{2} + F_3 l\frac{\sqrt{2}}{2} - F_4 l\frac{\sqrt{2}}{2} \\
\tau_y^B &= F_1 l\frac{\sqrt{2}}{2} + F_2 l\frac{\sqrt{2}}{2} - F_3 l\frac{\sqrt{2}}{2} - F_4 l\frac{\sqrt{2}}{2} \\
\tau_z^B &= -M_1 + M_2 - M_3 + M_4
\end{aligned} \tag{3.22}$$

$F_i = b\omega_i^2$ and $M_i = d\omega_i^2$ (Reaction moment), For this Thesis assuming that the propellers CW direction is Positive and CCW direction is Negative.

$$\begin{aligned}
\tau_x^B &= \frac{\sqrt{2}}{2} lb\left(\omega_1^2 + \omega_2^2 - \omega_3^2 - \omega_4^2\right) \\
\tau_y^B &= \frac{\sqrt{2}}{2} lb\left(\omega_1^2 - \omega_2^2 - \omega_3^2 + \omega_4^2\right) \\
\tau_z^B &= -M_1 + M_2 - M_3 + M_4 = -d\omega_1^2 + d\omega_2^2 - d\omega_3^2 + d{\omega_4}^2
\end{aligned} \tag{3.23}$$

from Equation (3.23)Let: $\tau_x^B = U_2 =$ Roll, $\tau_y^B = U_3 =$ Pitch and $\tau_z^B = U_4 =$ Yaw

Gyroscopic effects The overall balance will occur when the total rotor speeds expressed algebraically equals zero, as two of the propellers (propellers 2 and 4) rotate CW and the other two (propellers 1 and 3) rotate CCW. Otherwise, unbalance may occur. A gyroscopic effect could result from this imbalance. Moreover, the roll and pitch rates are proportionate to this effect. The propellers' overall speed is defined by the equation below.

$$F_{gyro}^B = -\sum_{k=1}^{4} Jr\left(\begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}\right)(-1)^k \omega_k \tag{3.24}$$

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} q \\ -p \\ 0 \end{bmatrix} \tag{3.25}$$

$$F_{gyro}^B = -\sum_{k=1}^{4} Jr\left(\begin{bmatrix} q \\ -p \\ 0 \end{bmatrix}\right)(-1)^k \omega_k \tag{3.26}$$

$$F_{9yro}^{B} = -Jr\left(\begin{bmatrix} -q \\ p \\ 0 \end{bmatrix}\omega_1 + \begin{bmatrix} q \\ -p \\ 0 \end{bmatrix}\omega_2 + \begin{bmatrix} -q \\ p \\ 0 \end{bmatrix}\omega_3 + \begin{bmatrix} q \\ -p \\ 0 \end{bmatrix}\omega_4\right) \tag{3.27}$$

$$F_{\text{gyro}}^{B} = -Jr\left(\begin{bmatrix} -q & q & -q & q \\ p & -p & p & -p \\ 0 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix}\right) \tag{3.28}$$

$$F_{gyro}^{B} = Jr\left(\begin{bmatrix} q & -q & q & -q \\ -p & p & -p & p \\ 0 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix}\right) \tag{3.29}$$

Let $w_r = \omega_1 - \omega_2 + \omega_3 - \omega_4$

$$F_{gyro}^{B} = Jr\left(\begin{bmatrix} q\omega_1 - q\omega_2 + q\omega_3 - q\omega_4 \\ p\omega_1 + p\omega_2 - p\omega_3 + p\omega_4 \\ 0+0+0+0 \end{bmatrix}\right) \tag{3.30}$$

$$F_{gyro}^{B} = Jr\left(\begin{bmatrix} q\left(\omega_1 - \omega_2 + \omega_3 - \omega_4\right) \\ p\left(-\omega_1 + \omega_2 - \omega_3 + \omega_4\right) \\ 0 \end{bmatrix}\right) \tag{3.31}$$

$$F^{B}{}_{gyro} = Jr\left(\begin{bmatrix} q\left(w_r\right) \\ p\left(-w_r\right) \\ 0 \end{bmatrix}\right) \tag{3.32}$$

Euler equation: Euler equation: In an IFF, the time rate of change of AM L of any arbitrary portion of a continuous body is equal to the total enjected torquesM acting on the plant. This is Euler's second axiom of Newton's second law, here it be called as the way of balancing torques. $\sum \tau^E = M^E = \left(\frac{dL}{dt}\right)_E$ AM((L) $= I\omega$ where: $I = mR^2$ and R is radius from rotation axis Torque $= I\alpha \approx I = ma$, I is the inertia.

$$\left(\frac{dL}{dt}\right)_E = \frac{I_B w}{dt} \tag{3.33}$$

Moment of Inertia (Rotational Inertia): Moment of I, also known as Rotational Moment of I, is the measure of how much a body resists angular acceleration. A large moment of inertia makes it difficult to rotate the mass (m). Knowing the moment of inertia makes it easier to estimate how challenging it might be to angularly accelerate the ISR Quadrotor. $I = mR^2$and Torque $= Ia$ **Inertia**: One useful method for condensing all of an ISR Quadrotor's moments of inertia into a single quantity is inertia. The inertia of a rigid body is represented as [27] :

$$J = \begin{bmatrix} \mathrm{I_{xx}} & 0 & 0 \\ 0 & \mathrm{Iyy} & 0 \\ 0 & 0 & \mathrm{Izz} \end{bmatrix} \tag{3.34}$$

Since quad-rotor geometry is symmetric, $\mathrm{I_{xy}} = \mathrm{I_{xz}} = \mathrm{I_{yx}} = \mathrm{I_{yz}} = \mathrm{I_{zx}} = \mathrm{I_{zy}} = 0$

Angular momentum

$$(L_E) = I_E \omega \tag{3.35}$$

$$\text{Angular momentum } (L_E) = I\omega = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix} = \begin{bmatrix} I_{xx}\omega_1 \\ I_{yy}\omega_2 \\ I_{zz}\omega_3 \end{bmatrix}_E \tag{3.36}$$

$$L_E = \begin{bmatrix} I_{xx}p \\ I_{yy}q \\ I_{zz}r \end{bmatrix}_E \tag{3.37}$$

$L_E = I_E w$:It is better to work in BFF because in the IFF, the body's orientation will be constantly changing, making it impossible to use a constant or FI; instead, it may need to be calculated constantly. For body frame, which can be expressed as a constant I times the AV, this makes sense.

$$\left(\frac{dL}{dt}\right)_{B-\text{ frame}} = \left(\frac{dL}{dt}\right)_{E-frame} + \vec{\omega} \times L \tag{3.38}$$

$$\tau^B = \dot{L} + \omega \times L \tag{3.39}$$

We refer to this as the Euler Equation.

$$\tau^B = \begin{bmatrix} I_{xx}\dot{p} \\ I_{yy}\dot{q} \\ I_{zz}\dot{r} \end{bmatrix} + \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} I_{xx}p \\ I_{yy}q \\ I_{zx}r \end{bmatrix} \tag{3.40}$$

$$\begin{bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} = \begin{bmatrix} I_{xx}\dot{p} \\ I_{yy}\dot{q} \\ I_{zz}\dot{r} \end{bmatrix} + \begin{bmatrix} rqI_{zz} - rqI_{yy} \\ rpI_{xx} - rpI_{zx} \\ pqI_{yy} - pqI_{xx} \end{bmatrix} = \begin{bmatrix} I_{xx}\dot{p} + rq\left(I_{zz} - I_{yy}\right) \\ I_{yy}\dot{q} + rp\left(I_{xx} - I_{zz}\right) \\ I_{zz}\dot{r} + pq\left(I_{yy} - I_{xx}\right) \end{bmatrix} \tag{3.41}$$

The motion generated with respect to the BF and the total torque τ applied at the BFF are implied by the equation (3.42). [27] [29].

$$\sum M^B = M^B_{\text{th}} + F^B_{\text{gyro}} = \dot{\vec{L}} + \vec{\omega} \times L \tag{3.42}$$

$$\underbrace{\begin{bmatrix} I_{xx}\dot{p} + rq\left(I_{xz} - I_{yy}\right) \\ I_{yy}\dot{q} + rp\left(I_{xx} - I_{xx}\right) \\ I_{zx}\dot{r} + pq\left(I_{yy} - I_{xx}\right) \end{bmatrix}}_{\dot{L}+\dot{\omega}\times L} = \underbrace{\begin{bmatrix} U_2 \\ U_3 \\ U_4 \end{bmatrix}}_{M^B_{\text{Thrust}}} + \underbrace{Jr\left(\begin{bmatrix} q\left(w_r\right) \\ p\left(-w_r\right) \\ 0 \end{bmatrix}\right)}_{F^B_{\text{gyro}}} \tag{3.43}$$

The following is the format of the dynamics motions of equation for The BFF frame defines the rotational motion.

$$\begin{aligned}\dot{p} &= \frac{U_2}{I_{xx}} - \frac{qr\left(I_{xx} - I_{yy}\right)}{I_{xx}} + \frac{Jrw_r q}{I_{xx}} \\ \dot{q} &= \frac{U_3}{I_{yy}} - \frac{rp\left(I_{xx} - I_{xx}\right)}{I_{yy}} - \frac{Jrw_r p}{I_{yy}} \\ \dot{r} &= \frac{U_4}{I_{xz}} - \frac{pq\left(I_{yy} - I_{xx}\right)}{I_{xx}}\end{aligned} \tag{3.44}$$

Using the transfer matrix as shown below, The description of the dynamics of the Earth's Inertial Reference Frame is acquired.:

$$\begin{bmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} = T \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} \tag{3.45}$$

$$\begin{aligned}\ddot{\phi} &= \dot{p} + \dot{q}(\sin(\phi)\tan(\theta)) + \dot{r}(\cos(\phi)\tan(\theta)) \\ \ddot{\theta} &= \dot{q}(\cos(\phi)) - \dot{r}(\sin(\phi)) \\ \ddot{\psi} &= \dot{q}(\sin(\phi)\sec(\theta)) + \dot{r}(\cos(\phi)\sec(\theta))\end{aligned} \tag{3.46}$$

The ISR Quadrotor's overall dynamics consist of four control inputs, or under-actuated dynamics, for the quadrotor: $\begin{bmatrix} U_1 & U_2 & U_3 & U_4 \end{bmatrix}^T$ The following dynamic equations represent U_1, the total thrust force (also known as the throttle) applied to the quadrotor body; U_2, the resultant torque affecting the roll angle of the ISR quad-rotor; U_3, the resultant torque affecting the pitch angle of the quad-rotor; and U_4, the resultant torque affecting the heading angle of the ISR quad-rotor UAV.

$$\begin{aligned}\ddot{x} &= (\cos(\phi)\sin(\theta)\cos(\psi) + \sin(\phi)\sin(\psi))\frac{U_1}{m} \\ \ddot{y} &= (\cos(\phi)\sin(\theta)\sin(\psi) - \sin(\phi)\cos(\psi))\frac{U_1}{m} \\ \ddot{z} &= (\cos(\phi)\cos(\theta))\frac{U_1}{m} - g \\ \ddot{\phi} &= \frac{U_2}{I_{xx}} - \frac{(I_{xx} - I_{yy})}{I_{xx}}\dot{\theta}\dot{\psi} + \frac{J_r\dot{\theta}w_r}{I_{xx}} \\ \ddot{\theta} &= \frac{U_3}{I_{yy}} - \frac{(I_{xx} - I_{xx})}{I_{yy}}\dot{\phi}\dot{\psi} - \frac{J_r\dot{\phi}w_r}{I_{yy}} \\ \ddot{\psi} &= \frac{U_4}{I_{xz}} - \frac{(I_{yy} - I_{xx})}{I_{xz}}\dot{\phi}\dot{\theta}\end{aligned} \tag{3.47}$$

The desired Roll (ϕ) and Pitch (θ) angles from the translational dynamics equation as below:

$$\begin{aligned}\phi_r &= \tan^{-1}\left(\frac{\cos(\theta_d)}{\ddot{z} + g}(\ddot{x}\sin(\psi) - \ddot{y}\cos(\psi))\right) \\ \theta_r &= \tan^{-1}\left(\frac{1}{\ddot{z} + g}(\ddot{x}\cos(\psi) + \ddot{y}\sin(\psi))\right)\end{aligned} \tag{3.48}$$

The control inputs will be used to calculate the rotor's angular velocity, and its inverse

relationship is defined as follows:

$$\begin{aligned}
\omega_1 &= \sqrt{\frac{U_1}{4b} + \sqrt{2}\frac{U_2}{4bl} + \sqrt{2}\frac{U_3}{4bl} + \frac{U_4}{4d}} \\
\omega_2 &= \sqrt{\frac{U_1}{4b} + \sqrt{2}\frac{U_2}{4bl} - \sqrt{2}\frac{U_3}{4bl} - \frac{U_4}{4d}} \\
\omega_3 &= \sqrt{\frac{U_1}{4b} - \sqrt{2}\frac{U_2}{4bl} - \sqrt{2}\frac{U_3}{4bl} + \frac{U_4}{4d}} \\
\omega_4 &= \sqrt{\frac{U_1}{4b} - \sqrt{2}\frac{U_2}{4bl} + \sqrt{2}\frac{U_3}{4bl} - \frac{U_4}{4d}}
\end{aligned} \tag{3.49}$$

The mathematical modeling phase has been completed, and the subsequent step is to verify the model. To accomplish this, the mathematical equation was programmed in MATLAB®SIMULINK software using parameters from literature sources. This was done in order to determine if the system accurately represents the real system. The following section goes into detail about the model verification process, specifically examining various flight conditions. A 3D simulation was employed to visualize the response of the plant to applied input, utilizing the Unreal 3D simulation from the UAV toolbox. The entire simulation was verified using MATLAB®2021a SIMULINK software. The mathematical model was developed and verified by comparing the modeled plant dynamics with the actual Quadrotor plant dynamics using unreal simulation, applying input U under different flight conditions.

3.3 Model Verification

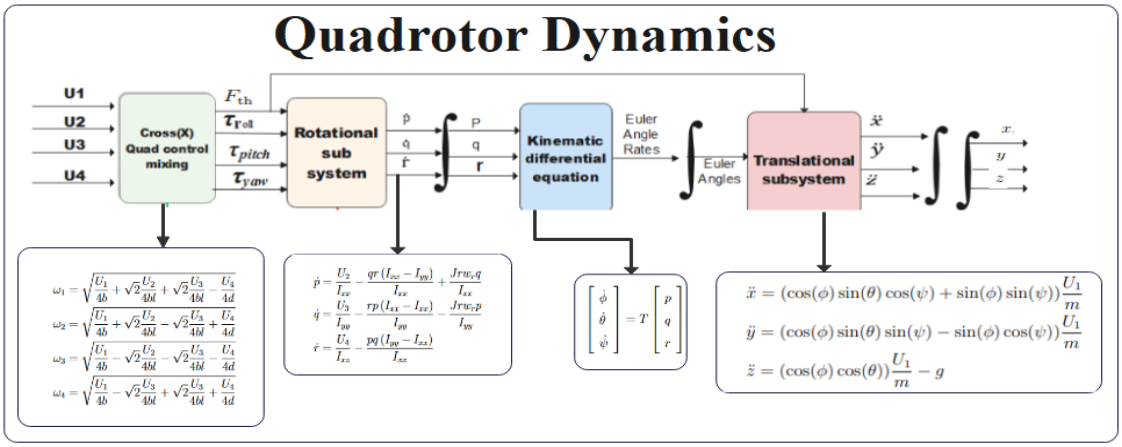


Figure 3.5: Model of Quad-Rotor Dynamics

Figure 3.6: Complete Model of Quad-Rotor Dynamics

Table 3.1: Parameters Description.

Parameters	Description	Units
$\begin{bmatrix} x & y & z \end{bmatrix}$	Linear position vector	m
$\begin{bmatrix} \phi & \theta & \psi \end{bmatrix}$	Angular position vector	rad
$\begin{bmatrix} u & v & w \end{bmatrix}$	Linear Velocity vector	m/s
$\begin{bmatrix} p & q & r \end{bmatrix}$	Angular velocity vector	rad/s
$\begin{bmatrix} I_{xx} & I_{yy} & I_{zz} \end{bmatrix}$	Moment of inertia vector	$\text{kg} \cdot \text{m}^2$
F_{th}	Total thrust generated by rotors	N
$\begin{bmatrix} \tau_x & \tau_y & \tau_z \end{bmatrix}$	Control torques	N.m
g	Gravitational force	m/s^2
m	Total mass	kg
$\begin{bmatrix} \omega_1 & \omega_2 & \omega_3 & \omega_4 \end{bmatrix}$	Rotors speeds vector	rad/s
b	Thrust coefficient	N.s s^2
l	Motor to center length	m
w_r	residual rotor speed	rad/s
d	Drag coefficient	N.m.s.s
Jr	rotor inertia vector	$\text{kg} \cdot \text{m}^2$

Table 3.2: Parameters Values [30]

Parameter	Value
I_{xx}	8.5532×10^{-3}
I_{yy}	8.5532×10^{-3}
I_{zz}	1.476×10^{-2}
g	9.81×10^{0}
m	1×10^{0}
b	7.66×10^{-5}
d	5.63×10^{-6}
l	2.2×10^{-1}
Jr	0.1×10^{-3}

3.3.1 Hovering Verification

Equilibrium Conditions for hovering of ISR Quadrotor are as below

$$mg = F_1 + F_2 + F_3 + F_4 = U_1$$

When $U_1 = F_{th} = mg, U_2 = U_3 = U_4 = 0$,plant response =hovering verification at this condition the drone stays hovering at specified altitude. When the total thrust force is equal to the quad-copter takeoff weight multiplied by gravitational acerelation at that case the quad-rotor stayed in hovering condition in this case the mass of plant was 1kg and gravitational acceleration was taken as 9.81 and the plant is stayed hovering as seen in the figure (3.7).

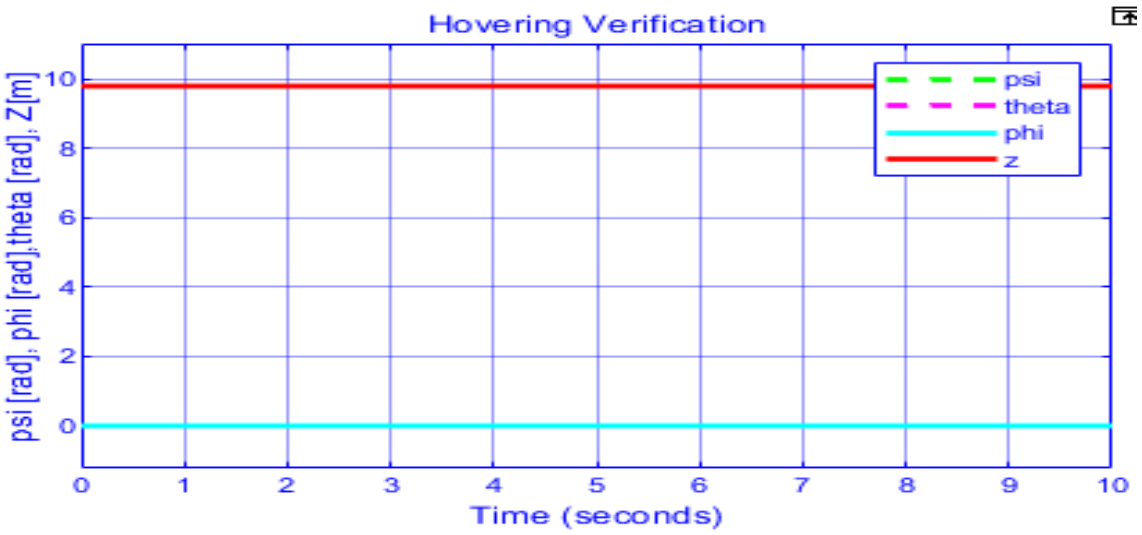


Figure 3.7: Hovering Verification

3.3.2 Altitude Verification(Throttle up)

Condition For Rise Motion

$$mg < U_1 \tag{3.50}$$

$U_1 > mg, U_2 = U_3 = U_4 = 0$ altitude verification(rise altitude)

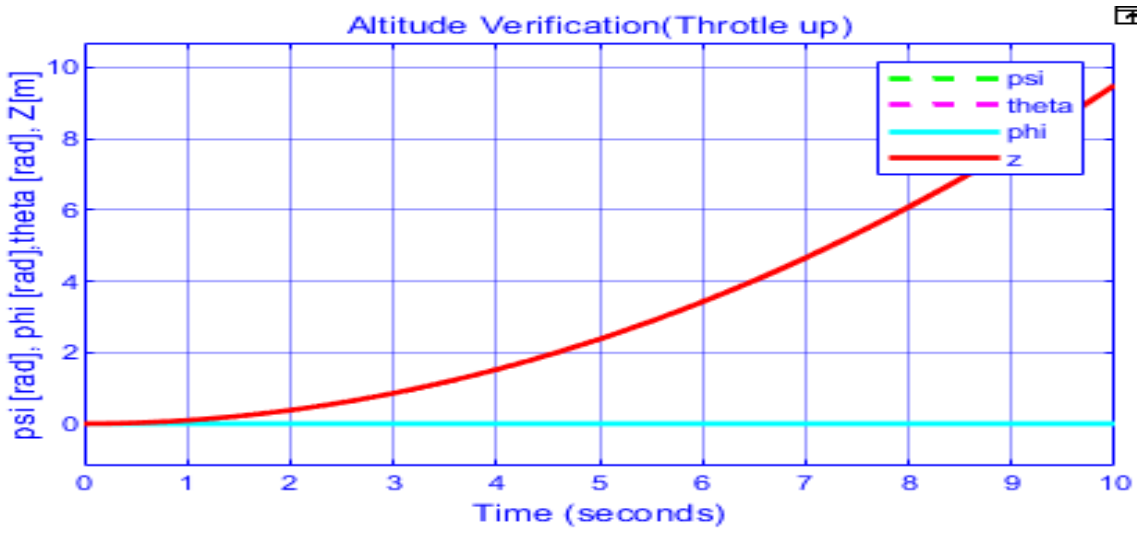


Figure 3.8: Drone Rise up Altitude Verification

Condition for Fall Motion

$$mg > U_1 \tag{3.51}$$

3.3.3 Roll Verification

When applying the roll torque which is 0.0001Nm and the pitch and yaw torque equal to zero and the plant stayed at hoovering condition and starts to roll at x axis.

$$U_1 = mg, U_2 = 0.0001Nm, U_3 = U_4 = 0$$

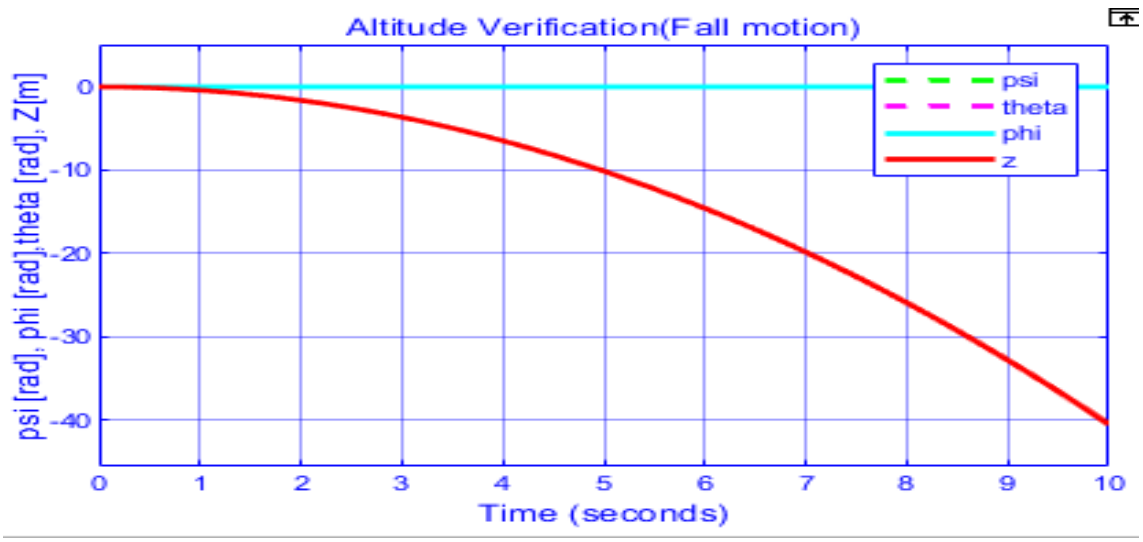


Figure 3.9: Drone Fall down Altitude Verification

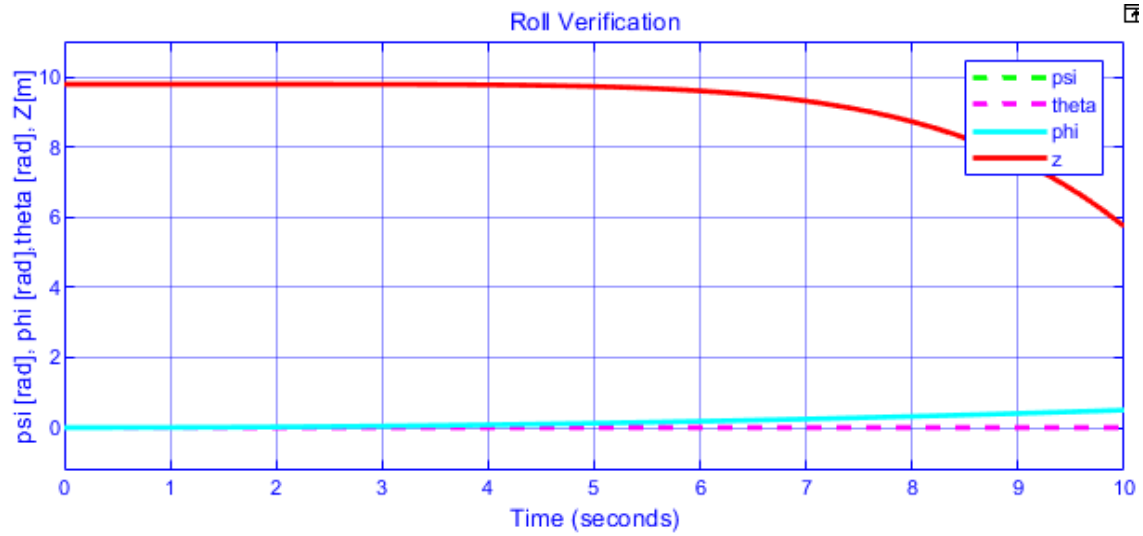


Figure 3.10: Roll Verification

3.3.4 Pitch Verification

When applying the pitch torque which is 0.0001Nm and the roll and yaw torque equal to zero and the plant stayed at hoovering condition and starts to pitch at y axis. When

$$U_1 = mg, U_3 = 0.0001Nm, U_2 = U_4 = 0$$

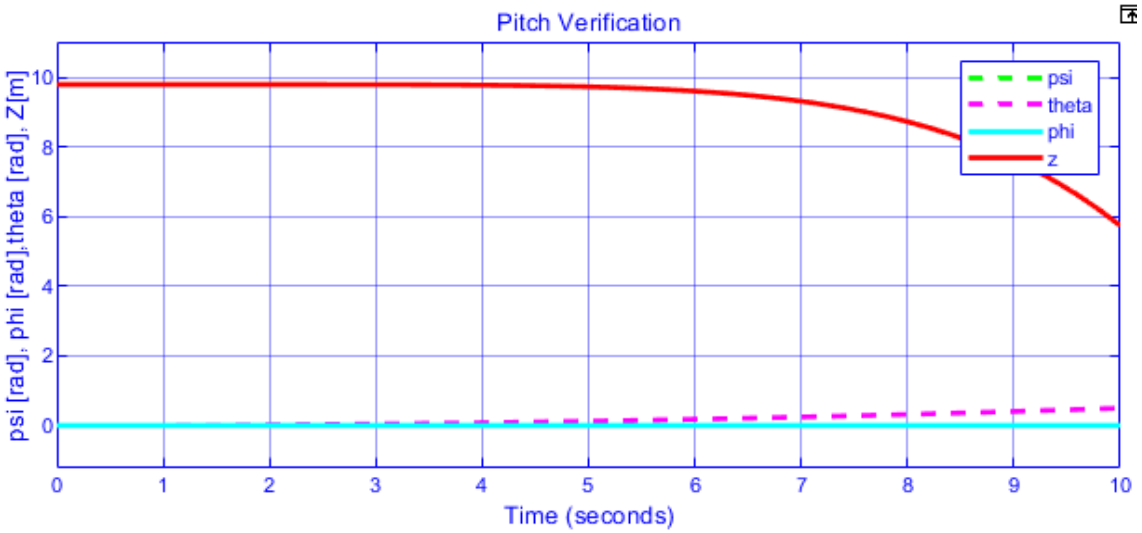


Figure 3.11: Pitch Verification

3.3.5 Yaw Verification

When applying the yaw torque which is 0.0001Nm and the roll and pitch torque equal to zero and the plant stayed at hoovering condition and starts to yaw at z axis.When

$$U_1 = mg, U_4 = 0.0001Nm, U_2 = U_3 = 0$$

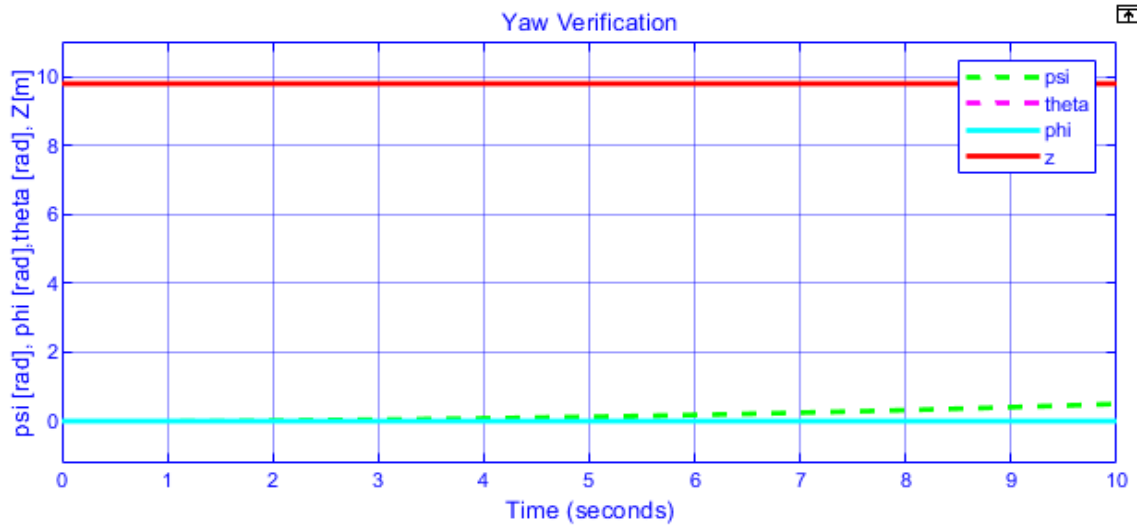


Figure 3.12: Yaw Verification

Therefore, the model is verified and accurately represents a real-world plant, which has been confirmed through the use of UAV toolbox unreal simulation. As a result, the next phase involves the development of a control algorithm to direct the quad-rotor along a predetermined trajectory. Instead of relying on control input as a verification case, the focus now shifts towards designing a control law for autonomous command. The design of the controller is a vital step in achieving autonomous control of the proposed plant, the quad-rotor. Consequently, the following chapter will delve into the design of controllers specifically tailored for the modeled ISR Quadrotor plant.

Chapter 4

Controller Design for Quad-Rotor

4.1 Introduction

This chapter focuses on the design of controllers for a quad-rotor UAV, specifically the ISR Quad-Rotor described in chapter three. Two types of controllers, GA Optimized Fuzzy-PID Controller and GA Optimized Fuzzy-PID Based Neural Network Controller, are developed to control the nonlinear model of the quad-rotor. The design of GA Optimized Fuzzy-PID Based Neural Network Controller using GA Optimized Fuzzy-PID Controller training data is an advanced method for creating intelligent control systems for quad-rotor.

Neural Network Controllers: Neural Network controllers are artificial intelligence-based controllers that can learn and adapt to control systems' dynamics. These controllers consist of interconnected nodes (neurons) organized into layers. Each neuron performs computations and transfers information to subsequent layers. Neural Network controllers can learn from training data to make predictions or control decisions based on input-output patterns.

GA Optimized Fuzzy-PID Controller: The GA Optimized Fuzzy-PID Controller used to get input output data and it combines fuzzy logic-based control with a PID controller and a Genetic Algorithm for optimization. This algorithm aims to enhance the optimal and robustness of the controller. Here's an overview of how it works:

Fuzzy Logic: Fuzzy logic allows handling imprecise or uncertain data by using linguistic variables. It uses fuzzy sets and fuzzy rules to define the relationships between inputs and outputs.

PID Control:PID controllers are widely used in control systems. They consist of three components - Proportional, Integral, and Derivative. The PID controller adjusts the control signal based on the error between the desired and actual output values.

Genetic Algorithm: Genetic Algorithms mimic the process of natural selection to optimi ze control parameters. It involves generating a population of potential solutions and using genetic operators like mutation and crossover to evolve and improve the solutions over multiple generations.

Combining these three elements, the GA Optimized Fuzzy-PID Controller utilizes fuzzy logic to define the control rules and map inputs to outputs. It then optimizes the control parameters using the Genetic Algorithm approach to achieve better control performance. In this chapter, two control laws were designed to control the proposed ISR Quad-rotor. The first control law involved the design of the GA Optimized Fuzzy-PID controller, while

the second control law utilized the input-output data generated from the GA Optimized Fuzzy-PID controller to design an GA Optimized Fuzzy-PID Based Artificial Neural Network (Fuzzy-PID based ANN) controller. The ANN controller was trained using the GA Optimized Fuzzy-PID Controller training data, making use of the MATLAB® NN toolbox.

4.2 GA Optimized Fuzzy-PID Controller Design

GA Optimized Fuzzy-PID Controller was designed to generate input output data for proposed controller and the control law was identical for six dynamics of the proposed plant(ISR Quadrotor). In GA Optimized Fuzzy-PID controller the PID gain is tuning by fuzzy rules and the scaling PID and PD values are tuned using Genetic Algorithm optimization method(GA). Its advantage is to reduce overshoot and track the reference trajectory within short time. The design of a GA Optimized Fuzzy-PID controller involves combining the concepts of fuzzy logic and PID (Proportional-Integral-Derivative) control to create an optimal and robust controller. The GA Optimized Fuzzy-PID Controller encompasses additional components beyond the conventional PID controller, including the use of a genetic algorithm (GA) for online learning and adaptation.

The GA Optimized Fuzzy-PID controller design starts by defining linguistic variables and membership functions that represent the input and output parameters of the controller. These linguistic variables are used to create fuzzy rules that govern the controller's behavior.

The PID controller component calculates control actions based on the error (the difference between the desired setpoint and the actual value) and its integral and derivative terms. The fuzzy logic component enhances the controller's adaptability by using linguistic variables and fuzzy rules to adjust the control actions based on the current system state.

To further optimize the performance of the Fuzzy-PID controller, a genetic algorithm is employed. The GA performs online learning by adjusting the parameters of the fuzzy logic control system based on a predefined fitness function. Through multiple iterations, the GA searches for an optimal set of parameters that minimizes the error and achieves desired control performance.

The GA Optimized Fuzzy-PID controller design offers several advantages. By incorporating fuzzy logic, the controller can handle non-linearities and uncertainties in the system. The integration of the genetic algorithm enables the controller to improve its performance over time. This flexibility and self-tuning capability make the FPID-GA controller suitable for a wide range of applications. Careful consideration should also be given to the selection of appropriate membership functions, fuzzy rules, and the fitness function for the genetic algorithm to ensure optimal control performance.

Overall, the GA Optimized Fuzzy-PID Controller design offers a promising approach to control systems that require robustness and optimal performance. Its ability to combine fuzzy logic, PID control, and genetic algorithms sets it apart from conventional controllers and makes it a valuable tool in various engineering applications.

4.2.1 Fuzzy System for Tuning the PID Gains Online

The approach proposed in this thesis is to use a fuzzy system for online tuning of PID gains in the control algorithm of an ISR quadrotor. This is necessary because the quadrotor is highly nonlinear and underactuated, making classical PID control inadequate.

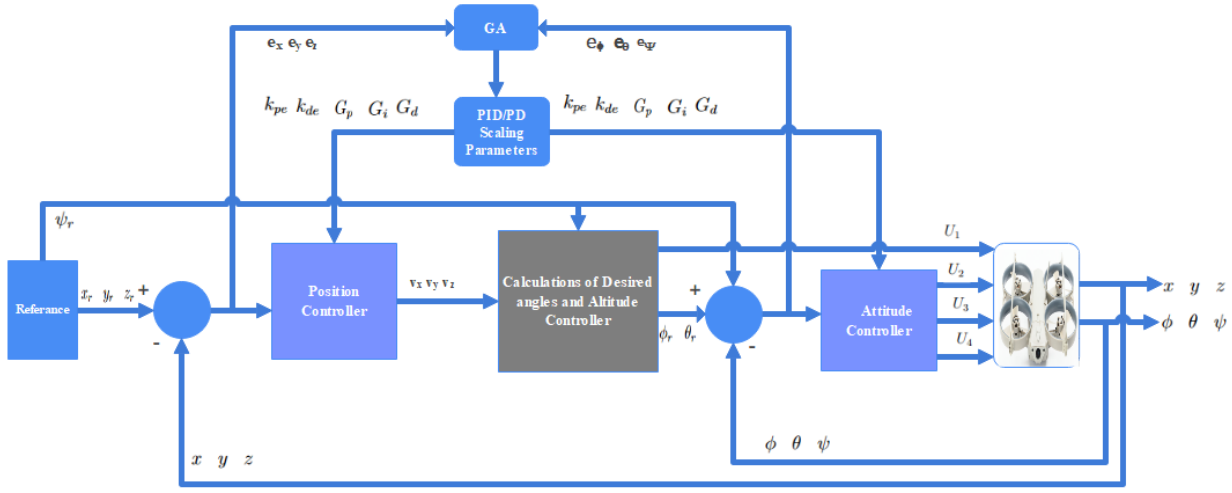


Figure 4.1: Control scheme Block Diagram

The first step in this approach is to establish a set of tuning rules or fuzzy IF-THEN rules for the PID gains. These rules are designed based on the specific requirements and characteristics of the quadrotor system. The rules determine how the PID gains should be adjusted in response to various inputs and conditions.

Once the tuning rules are defined, they are combined into a fuzzy system that is capable of adjusting the PID gains on-the-fly. This fuzzy system utilizes the inputs from the quadrotor's feedback signals to determine the appropriate changes in the PID gains for better control performance.

To optimize the performance of the fuzzy system, a genetic algorithm optimization (GA) is used to automatically tune the scaling factors associated with the PID parameters. This allows for an optimal adjustment of the PID gains based on the current operating conditions of the quadrotor.

By implementing the fuzzy system for online PID parameter gain tuning, the control algorithm of the ISR quadrotor can optimize its performance in fly, accommod ating for the system's nonlinearity and underactuated nature. This approach offers a more effective and robust control strategy compared to traditional PID control [31].

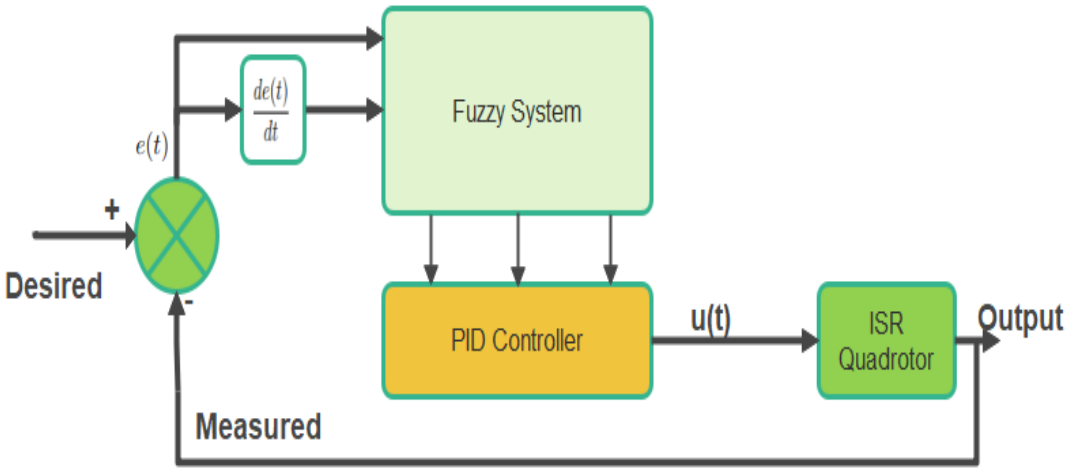


Figure 4.2: Basic Structure of Plant(ISR Quadrotor) with Fuzzy-PID

The concept of self-tuning refers to the ability of a fuzzy controller to automatically adjust its control parameters. This enables the controller to optimize the process output under varying operational conditions. In the case of a fuzzy self-tuning PID controller, the control rules are designed based on theoretical analysis and the expertise of human experts.

The controller uses the error "e" and its rate of change "de" as inputs. By applying fuzzy control rules, the controller can modify the proportional integral derivative gains automatically. This self-tuning process involves establishing a fuzzy relationship between "e", "de", and the three PID gains. Based on the fuzzy control algorithm, the gains are adjusted to meet the control requirements when there are variations in "e" and "de". This helps in achieving a good performance of the plant, specifically for ISR quad-rotors.

The scaling factors of the fuzzy proportional integral derivative controller are tuned using a Genetic Algorithm. This optimization technique ensures that the scaling factors are adjusted to improve the performance of the controller.

To control the defined reference trajectory tracking of the Mission quad-rotor, a combination of a proportional integral derivative controller and a fuzzy system is applied. The fuzzy logic system automatically calculates the proportional integral derivative gains based on fuzzy sets and rules. For more detailed information on the control structures of the GA Optimized Fuzzy-PID Controller for ISR quad-rotor, refer to figure (4.3) which provides a visual representation of the control structures.

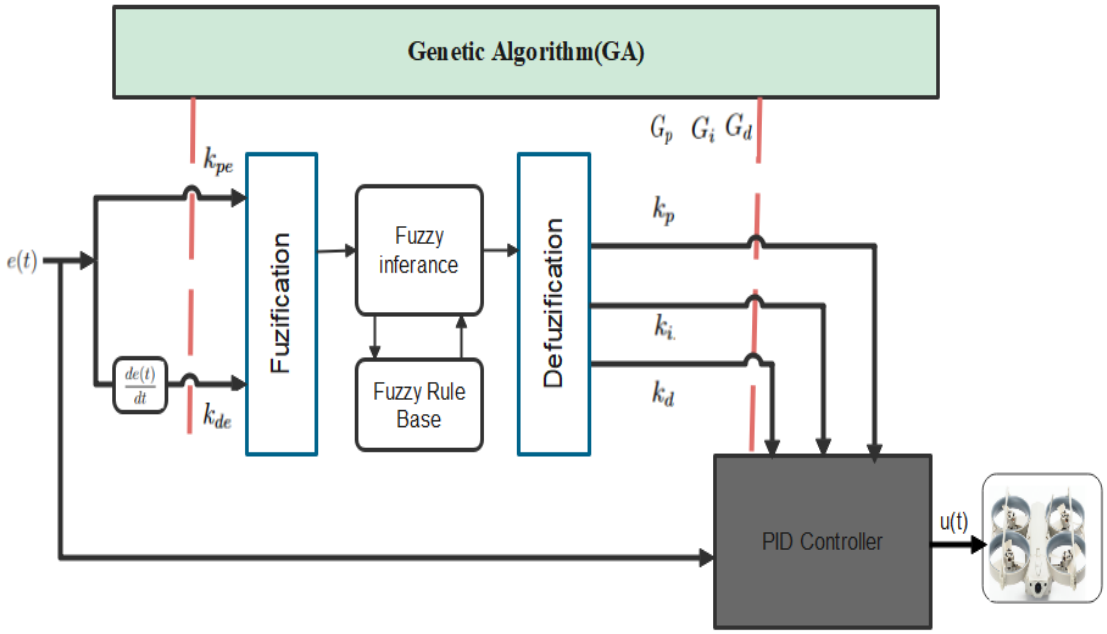


Figure 4.3: Structure of GA Optimized Fuzzy-PID Controller

$$\begin{aligned} e(t) &= \mathrm{r}(t) - y(t) \\ de(t) &= \frac{de(t)}{dt} \end{aligned} \tag{4.1}$$

where,r $= \begin{bmatrix} \phi_r & \theta_r & \psi_r & x_r & y_r & z_r \end{bmatrix}^T$ denotes the desired reference and y $= \begin{bmatrix} \phi & \theta & \psi & x & y & z \end{bmatrix}^T$ represents the output of the controlled ISR Quad-Rotor. The cost functions(fitness functions) of the GA optimization problem are chosen according to each controlled variable as the integral Time absolute error (ITAE) criteria.

The output variables of the optimized fuzzy tuning gains K_p, K_i, K_d are obtained by using the fuzzy reasoning inference(fuzzy rules) and the scaling factors G_p, G_i, G_d are obtained using Genetic optimization algorithm (GA). These variables combined and gives as the optimal proportional integral and derivative gains K_{p0}, K_{io} and K_{do} which are given respectively as follows:

$$\begin{cases} k_{po} = G_p k_p \\ k_{io} = G_i k_i \\ k_{do} = G_d k_d \end{cases} \tag{4.2}$$

where $k_p = \begin{bmatrix} k_{p\phi} & k_{p\theta} & k_{\psi p} & k_{px} & k_{py} & k_{pz} \end{bmatrix}^T$, $ki = \begin{bmatrix} k_{i\phi} & k_{i\theta} & k_{\psi i} & k_{ix} & k_{iy} & k_{iz} \end{bmatrix}^T$, and $k_d = \begin{bmatrix} k_{d\phi} & k_{d\theta} & k_{\psi d} & k_{dx} & k_{dy} & k_{dz} \end{bmatrix}^T$,denote the fuzzy inference reasoning(Fuzzy rule) outputs for proportional, integral and derivative actions of PID controllers, the parameters, $G_p = \begin{bmatrix} G_{p\phi} & G_{p\theta} & G_{p\psi} & G_{px} & G_{py} & G_{pz} \end{bmatrix}^T, G_i = \begin{bmatrix} G_{i\phi} & G_{i\theta} & G_{i\psi} & G_{ix} & G_{iy} & G_{iz} \end{bmatrix}^T, G_d = \begin{bmatrix} G_{d\phi} & G_{d\theta} & G_{d\psi} & G_{dx} & G_{dy} & G_{dz} \end{bmatrix}^T$, $k_{pe} = \begin{bmatrix} k_{pe\phi} & k_{pe\theta} & k_{pe}\psi & k_{pex} & k_{pey} & k_{pez} \end{bmatrix}^T$, $k_{de} = \begin{bmatrix} k_{de\phi} & k_{de\theta} & k_{de}\psi & k_{dex} & k_{dey} & k_{dez} \end{bmatrix}^T$ are the values of input and output scaling factors, introduced to eliminate the classical predefined ranges on parameters of the Fuzzy-PID/PD controllers [32] [33].

A set of linguistic rules was used in the fuzzy reasoning inference(sets) block to get the parameters K_p, K_i, K_d automatically [34] [35]. Seven fuzzy labels (Negative Big (NB), Negative Medium(NM), Negative Small (NS), Zero (ZO), Positive Small (PS), Positive Medium (PM)and Positive Big (PB)) [32]are used for the fuzzy input variables and seven fuzzy labels (very very small(VVS),very small(VS),Small (S), Medium (M),Big(B),Very big(VB)and very very big(VVB)) [32]for the fuzzy output variable. These linguistic variables are listed in Tables (4.1)and(4.2)which contain the 49 rules for the Fuzzy-PID controller. The membership functions for the input variables are defined with the triangular and the output is Gaussian.

Table 4.1: Fuzzy rules for K_d [34]

$de\backslash e$	NB	NM	NS	Z	PS	PM	PB
NB	M	B	VB	VVB	VB	B	M
NM	S	M	B	VB	B	M	S
NS	VS	S	M	B	M	S	VS
Z	VVS	VS	S	M	S	VS	VVS
PS	VS	S	M	B	M	S	VS
PM	S	M	B	VB	B	M	S
PB	M	B	VB	VVB	VB	B	M

For the Fuzzy logic control related to attitude and Position, the error is normalized to the interval $\begin{bmatrix} -1 & 1 \end{bmatrix}$, the error rate is confined within the range$\begin{bmatrix} -10 & 10 \end{bmatrix}$and the output is also normalized to the ranges $\begin{bmatrix} 0.2 & 0.7 \end{bmatrix}$, $\begin{bmatrix} 0.001 & 0.01 \end{bmatrix}$and $\begin{bmatrix} 0.1 & 0.15 \end{bmatrix}$ [32] [34] [36]for proportional,integral and derivative gains respectively.For the quad-copter positions and attitude Dynamic control requirement, the control system is identical and only the scaling factors are different depending on automatic tuning of Genetic Algorithm optimization and after defining the rule base, the next step is to determine the membership functions (MFs) for each parameter. These MFs are the same or identical for all variables. By employing the product-sum inference and center of gravity defuzzification method, the overall system

Table 4.2: Fuzzy rules for K_p andK_i [34]

$de\backslash e$	NB	NM	NS	Z	PS	PM	PB
NB	M	S	VS	VVS	VS	S	M
NM	B	M	S	VS	S	M	B
NS	VB	B	M	S	M	B	VB
Zo	VVB	VB	B	M	B	VB	VVB
PS	VB	B	M	S	M	B	VB
PM	B	M	S	VS	S	M	B
PB	M	S	VS	VVS	VS	S	M

outputs of each Fuzzy Logic Control can be obtained. However, in the case of quad-rotor dynamics, the trial and error approach for selecting fuzzy parameters may not be sufficient to achieve the necessary control actions due to their nonlinear behavior. In such situations, static values of scaling factors and single MFs alone cannot generate the desired control action for the system. To address this issue, a novel strategy is proposed to automatically design and fine-tune the input/output scaling factor parameters of the Fuzzy-PID controllers for the tracking control problem of a quad-copter UAV.

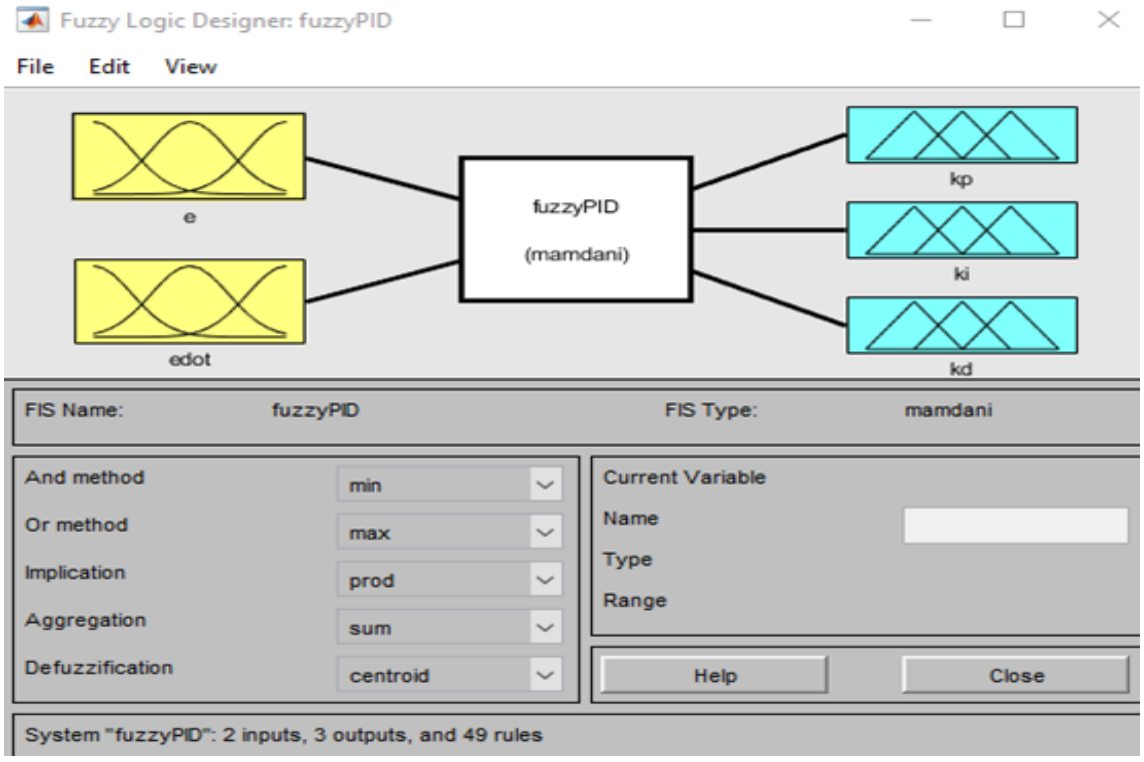


Figure 4.4: Fuzzy-PID 2 Inputs 3 Outputs and 49 rules

4.2.2 Fuzzy-PID/PD Gain Tuning

Hence ,the Fuzzy-PID controller is designed and the next work will be adjusting the scaling factors(PID/PD) gain values automatically using Genetic Algorithm Optimization Techniques as described above using the fitness functions as the integral time absolute error(ITAE).Then

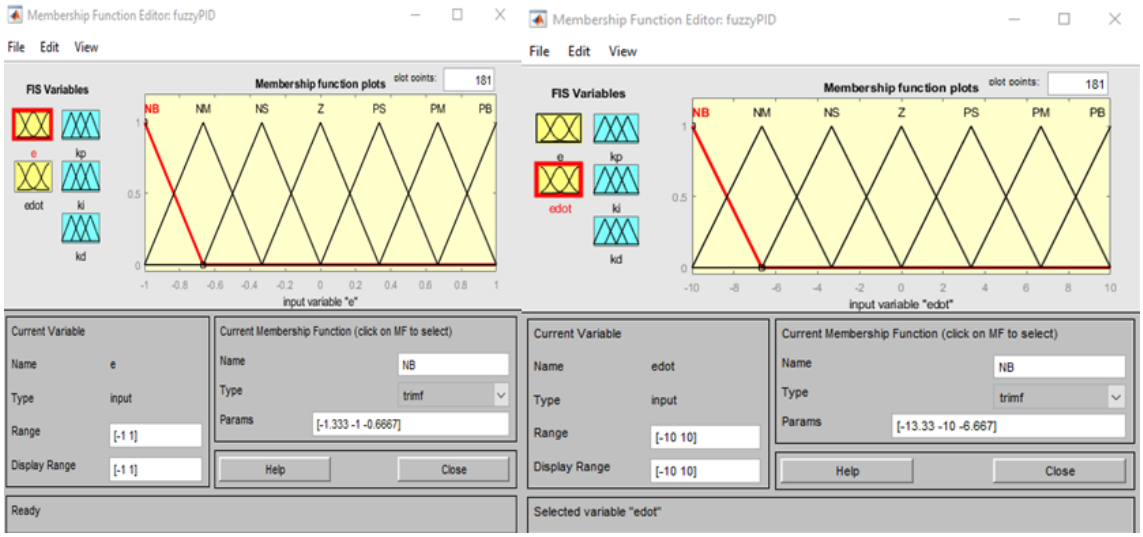


Figure 4.5: Inputs error and its rate Membership Function

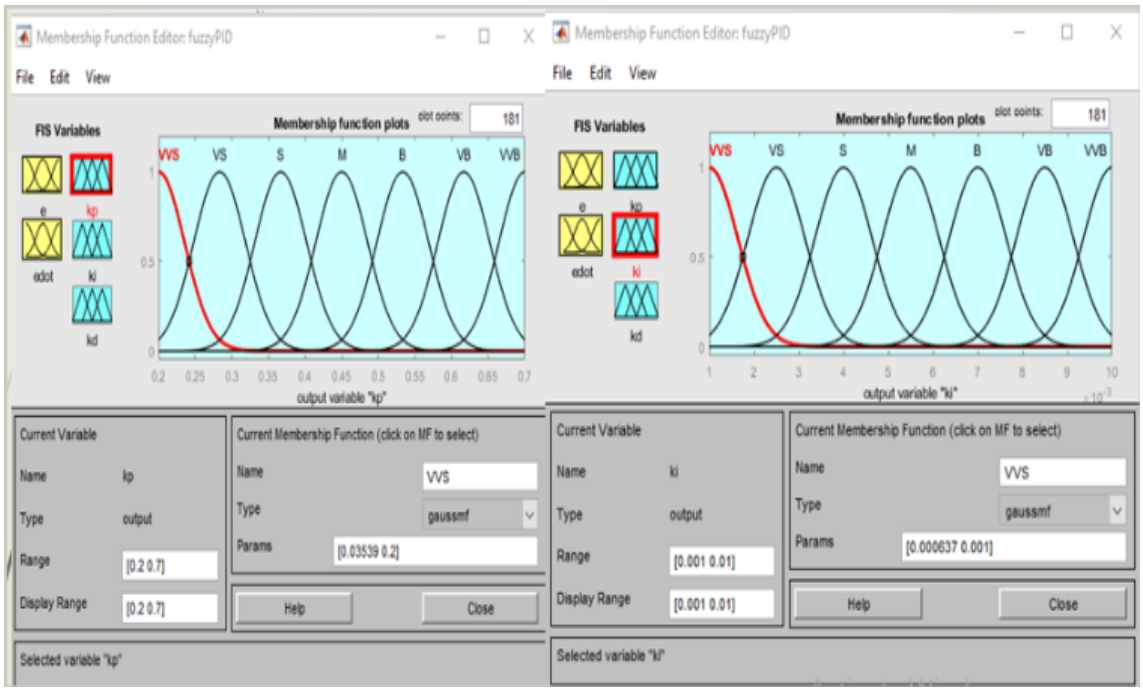


Figure 4.6: Output k_p and k_i Membership Function

the over all fuzzy controller outputs can be reformulated as below:

$$\begin{aligned} U_{FPID-GA}(t) &= G_p \times k_p(t) + G_i \times k_i \int_0^t e(t)dt + G_d \times k_d \frac{de(t)}{dt} \\ U_{FPID-GA}(t) &= k_{pO}e(t) + K_{iO} \int_0^t e(t)dt + K_{dO} \frac{de(t)}{dt} \end{aligned} \tag{4.3}$$

4.2.3 Genetic Algorithm Optimization (GA)

The settings of a genetic algorithm (GA) optimization can greatly influence its performance and convergence to an optimal solution. Here are some essential settings to consider when configuring a GA [37]:

Population Size: The population size determines the number of candidate solutions

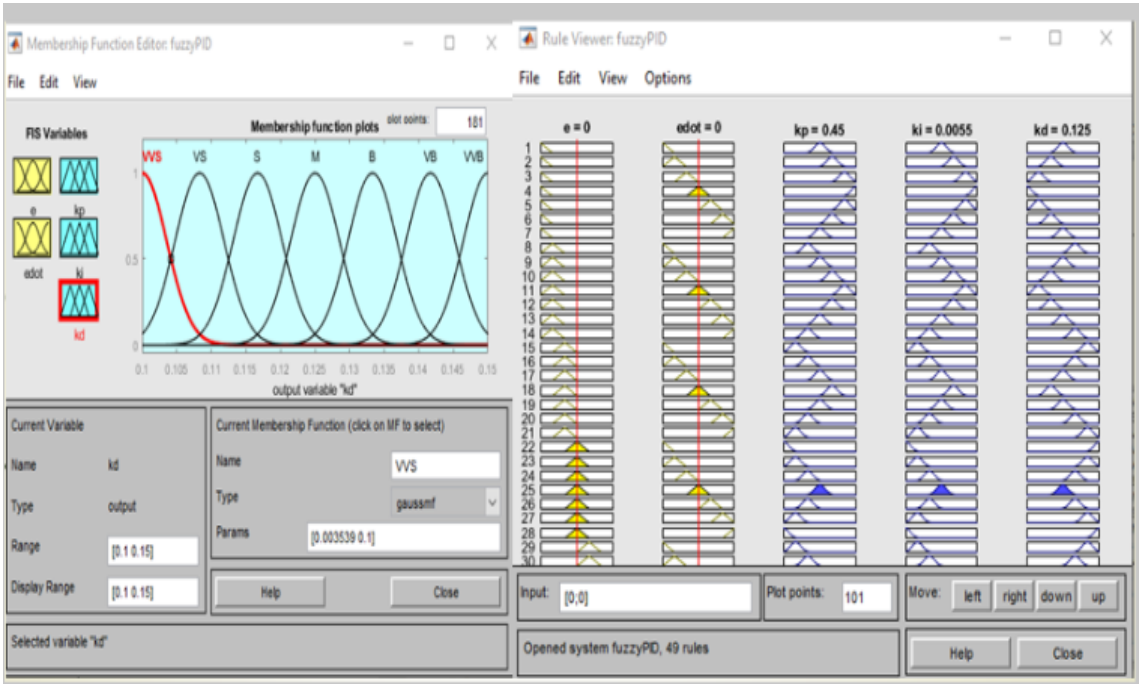


Figure 4.7: Output k_d Membership Function and over all Fuzzy-PID Rule Viewer

(individuals) in each generation. A larger population can help explore a wider search space but may increase computation time.

Initialization: The initial population can be generated randomly or based on some prior knowledge of the problem. Careful initialization can expedite convergence.

Crossover Rate: The crossover rate determines the probability of performing crossover (recombination) between two parent individuals to create new offspring. A higher crossover rate promotes exploration and exchange of genetic information.

Mutation Rate: The mutation rate defines the probability of introducing small random changes in an individual's genetic code. Mutation helps avoid becoming trapped in local optima and promotes exploration. **Fitness Function:** The fitness function quantifies the quality of each individual in the population. It evaluates how well an individual satisfies the objectives of the optimization problem. Defining an appropriate fitness function is crucial for GA convergence.

Termination Criteria: The termination criteria specify when to stop the optimization process. This can be based on a maximum number of generations, reaching a desired fitness level, or observing little improvement over successive generations.

It's worth mentioning that these settings 4.8 may need to be fine-tuned according to the problem at hand. Different problems may require different parameter combinations to achieve optimal results. Experimentation and understanding the problem domain are essential for identifying the most suitable settings.

4.2.3.1 Fitness Function

The fitness function or performance index is a quantification of how well the proposed plant (ISR Quad-rotor) responds to different inputs using an optimization technique. It is defined to reduce the discrepancies between the ISR quad-rotor's controlled and desired (commanded) output responses [37].

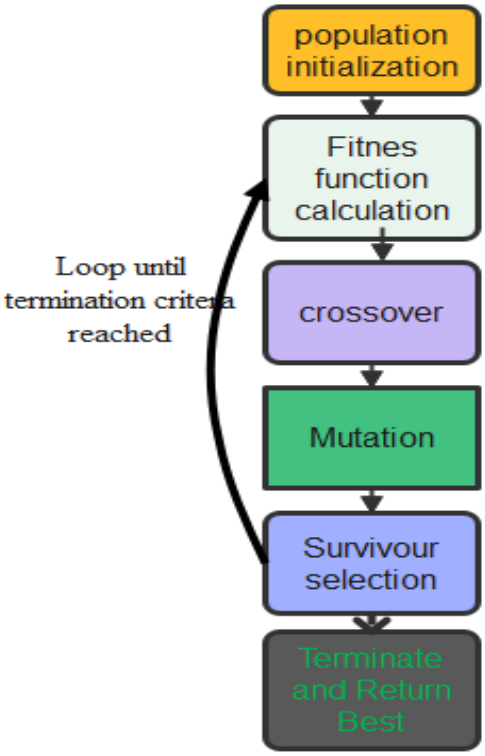


Figure 4.8: Flow Chart for GA Optimization

In this thesis, the fitness function is chosen to reduce the amount of error in the suggested plant trajectory tracking's position and orientation. The cost function or fitness function is derived from the errors of the measured and desired Mission Drone. According to Appendix A, the fitness functions that the discipline prefers.

A succinct description of every fitness function is given below:

The **ISE** index penalizes small errors less severely and large errors more severely. When a system is designed with this criterion in mind, it will strive to produce a fast and oscillatory response by rapidly decreasing a large initial error.

ITSE heavily penalizes errors that occur late in the system's transient response and emphasizes errors that occur early.

IAE penalizes control errors. It aims to minimize the accumulated absolute value of errors throughout the system's response.

ITAE penalizes long settling time and control errors. It aims to minimize the integral of the absolute value of errors over time. Systems designed using this criterion tend to have small overshoots and minimum damped oscillations.

In this thesis, the ITAE performance index is used as the cost function or fitness function to optimize the parameters of the FPID/PD values (scaling factors). The weight of ITAE values is set to unity to ensure equal importance is given to each term in the fitness function.

4.3 GA Optimized Fuzzy-PID Based Neural Network Controller(NNFPID-GA) Design

4.3.1 Introduction

Neural network controller design (NN) involves the development of artificial neural networks (ANNs) to control various systems or tasks. Here are some important considerations for designing a neural network controller:

Architecture: Determine the architecture of the neural network, including the number of layers, the number of neurons in each layer, and the connectivity pattern. Different architectures, such as feedforward networks, recurrent networks, or convolutional networks, may be suitable depending on the specific task.

Activation Functions: Select appropriate activation functions for the neurons in the network. Common choices include sigmoid, tanh, ReLU, or softmax functions. The activation function introduces non-linearities, allowing the network to learn complex relationships.

Training Data: Gather a diverse and representative dataset for training the neural network. The dataset should encompass different operating conditions and cover a wide range of inputs and outputs. Sufficient training data helps the network capture the underlying system dynamics.

Training Algorithm: Choose a training algorithm to update the network's weights and biases based on the input-output pairs from the training data. Popular algorithms include backpropagation, stochastic gradient descent (SGD), or variants like Adam or RMSprop. The choice depends on the complexity of the problem and the available computational resources.

Loss Function: Define an appropriate loss function that quantifies the discrepancy between the neural network's output and the desired output. The loss function guides the training process by providing a measure of the network's performance. Common loss functions include mean squared error (MSE), cross-entropy, or custom-defined functions.

Validation and Testing: After training the network, validate its performance using a validation dataset separate from the training data. This step helps assess the generalization capability of the network. Additionally, evaluate the network's performance on a testing dataset to obtain an unbiased estimate of its performance.

Iterative Refinement: Refine and optimize the neural network controller iteratively based on its performance. Fine-tune the network's architecture, training algorithm, hyperpara meters, or incorporate additional data to improve its effectiveness.

Evaluation Metrics: Choose appropriate metrics to evaluate the performance of the neural network controller. These metrics could be task-specific, such as mean squared error, accuracy, precision-recall, or any other relevant measure.

Remember that designing an effective neural network controller often requires a combinati on of expertise in neural networks, system dynamics, and the specific task or problem domain. Experimentation and continuous refinement play a crucial role in achieving desirable controller performance.

The Relationship Between Biological Neural Network(BNN) and Artificial Neural Network(ANN)

The biological neural network and artificial neural network are connected through the concept of modeling the structure and functionality of the human brain [38].

Biological neural networks are the fundamental building blocks of our nervous system. They consist of interconnected neurons that communicate through electrical and chemical signals. These networks are responsible for various cognitive functions and behaviors in living organisms.

On the other hand, artificial neural networks (ANNs) are computational models inspired by the structure and functioning of biological neural networks. ANNs are designed to simulate the learning and decision-making processes of the human brain, albeit in a much more simplified manner.

The relationship between biological and artificial neural networks lies in the abstraction and imitation of the brain's behavior. ANNs attempt to replicate the basic elements of biological neural networks, such as neurons and synapses, as well as the patterns of connectivity and information flow.

Although ANNs are not identical to biological neural networks, they can achieve similar tasks through the process of training. Just as the human brain learns from experience, ANNs learn from labeled datasets by adjusting the weights and biases of their artificial neurons. This process, known as training or learning, allows ANNs to recognize patterns, classify data, make predictions, and solve complex problems.

However, it is important to note that there are significant differences between biological and artificial neural networks. Biological neural networks are highly complex and dynamic systems, whereas artificial neural networks are more structured and fixed in design. Additionally, ANNs lack biological features like emotions, consciousness, and the ability to adapt to new situations in the same way as biological neural networks.

Nevertheless, the study of biological neural networks provides valuable insights and inspiration for the development and improvement of artificial neural networks [39] [40]. By understanding the underlying principles of biological neural networks, researchers can enhance the efficiency and capabilities ANN development, resulting in advancements in fields such as machine learning, artificial intelligence, and cognitive science.

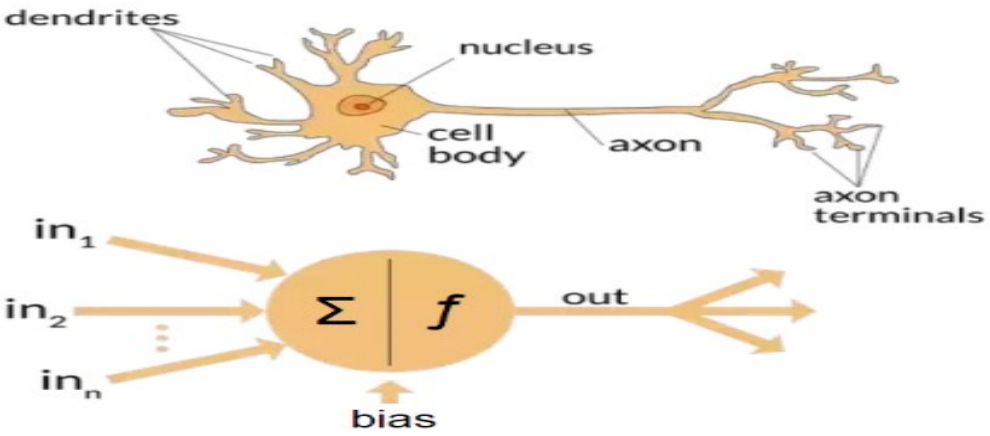


Figure 4.9: The Relationship between Biological Neural Network and Artificial Neural Network [41]

4.3.2 Basic Neural Network Model

A basic neural network model, also known as a feedforward neural network or a multilayer perceptron, consists of three main components: an input layer, hidden layers, and an output layer.

Input Layer: The input layer receives the input data for the neural network. Each input node represents a feature or attribute of the data. The number of input nodes is determined by the dimensionality of the input data.

Hidden Layers: The hidden layers are intermediate layers between the input and output layers. They consist of multiple nodes or neurons that perform computations on the input data. The number of hidden layers and the number of nodes in each layer can vary depending on the complexity of the problem being addressed. Each node in a hidden layer takes inputs from the previous layer and applies an activation function to produce an output.

Output Layer The neural network's final output, or prediction, is generated by the output layer. The kind of problem being solved determines how many nodes are in the output layer. For instance, in binary classification problems, the probability or confidence of belonging to a particular class would be represented by a single output node. For multiclass classification problems, there would be multiple output nodes, each representing the probability of belonging to a different class.

During training, the neural network learns the optimal weights and biases associated with each connection between nodes. This is achieved through an optimization algorithm such as gradient descent, which minimizes a loss function that quantifies the difference between the predicted outputs and the true labels of the training data. The basic neural network model can be further enhanced with techniques like regularization (e.g., dropout), activation functions (e.g., sigmoid, tanh, ReLU), initialization strategies (e.g., Xavier, He), and various optimization algorithms (e.g., Adam, RMSProp).

It's important to note that neural networks can be more complex and include additional layers or specialized architectures (e.g., convolutional neural networks for image processing, recurrent neural networks for sequence data) to address specific types of problems.

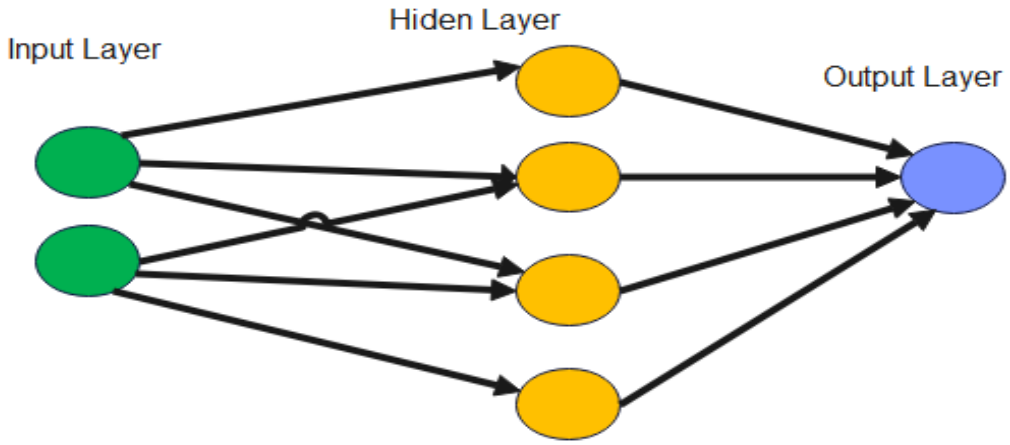


Figure 4.10: Feedforward Neural Network Architecture

The Model of Neural Network Can be described by a series of functional transformations

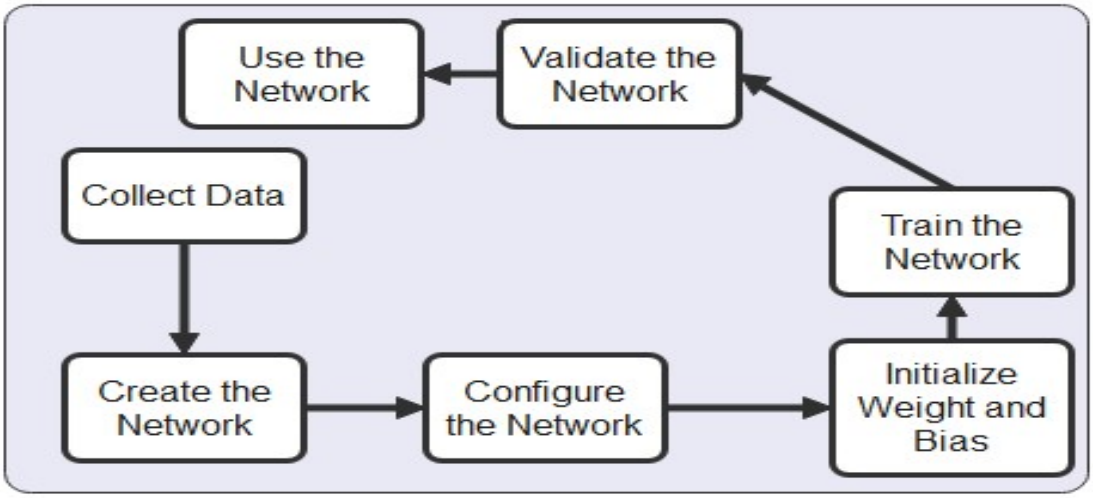


Figure 4.11: Work Flow of NN

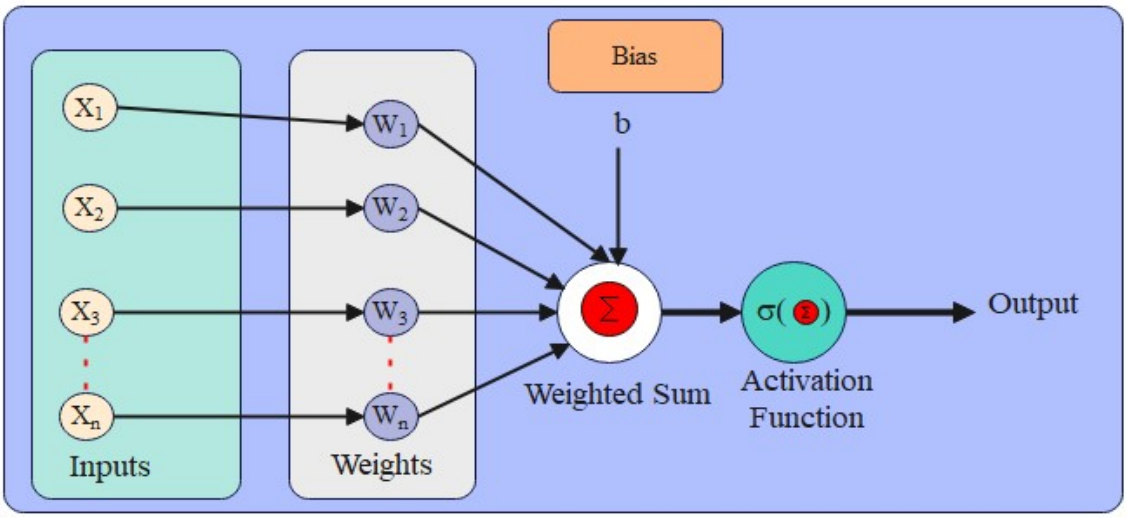


Figure 4.12: Jobs of Neural Network(Computation Algorithm)

N input variables $x_1, .., x_N$ M linear combinations in the form

$$J_j = \sum_{i=1}^{N} w_{ji}^{(1)} x_i + b_{j0}^{(1)} \quad \text{where } j = 1, .., M \tag{4.4}$$

Superscript (1) indicates parameters are in first layer of network
Parameters ${w_{ji}}^{(1)}$ are referred to as weights
Parameters ${b_{j0}}^{(1)}$ are biases, with $x_0 = 1$
Quantities J_j are known as output and x_i input to activation functions)

Activation Function

In this thesis for activation function the sigmoid function was used and each activationJ_j is transformed using differentiable nonlinear activation function as below.

$$E_j = \sigma\left(J_j\right) \tag{4.5}$$

WhereE_j first layer of network or hidden units

$$\sigma(J) = \frac{1}{1+e^{-J}} \tag{4.6}$$

Output $Y = \sigma\left(W_1^{\star}X_1 + W_2^{*}X_2 + \ldots. + W_n^{*}X_n + b\right)$ Where ,$\sigma(x) = \frac{1}{1+e^{-x}}$
Cost = (prediction-target)^2

$$MSE = (1/n) * \sum (y - \hat{y})^2$$

4.3.2.1 Levenberg Marquardt Algorithm-Based Backpropagation Learning Algorithm

Neural network training is commonly accomplished using the backpropagation learning algorithm. Its foundation lies in the idea of reducing the error between the training data's true labels and the network's predicted outputs [42].

The neural network's weights and biases are updated by the backpropagation algorithm through the use of gradient descent optimization. These parameters are adjusted iteratively in the direction of error minimization. The algorithm, which starts at the output layer and works its way backward through the network, is known as "backpropagation" because it computes the gradient of the error with respect to each weight in the network [43].

Levenberg-Marquardt algorithm is an improvement over the traditional gradient descent optimization [44] used in backpropagation. It combines the advantages of both the Gauss-Newton method and the steepest descent method to find a more efficient solution. By adapting the learning rate based on the curvature of the error surface, the Levenberg-Marquardt algorithm can converge faster and provide more accurate results.

In the context of backpropagation, the Levenberg-Marquardt algorithm adjusts the weights and biases by calculating the Hessian matrix [45], which represents the second-order derivatives of the error with respect to the weights. This matrix is then used to compute the update steps for the weights and biases, taking into account the curvature of the error surface.

By using a combination of gradient descent and the Levenberg-Marquardt algorithm, the backpropagation learning algorithm can effectively train neural networks to improve their prediction accuracy. It is especially useful when dealing with complex or highly nonlinear problems where traditional gradient descent may converge slowly or get stuck in local minima. All things considered, the Levenberg-Marquardt algorithm-based backpropagation learning algorithm is an effective tool for optimizing neural network performance during training.

4.3.2.2 Advantages of NN Controller

Advantages of using a neural network (NN) controller include:

Suitable for systems without mathematical models: NNs can be used as a "black box" model for plants when the mathematical models of plant dynamics are not available [46]. They can learn the system behavior from input-output data, making them effective in situations where only numerical information is available.

Adaptive control: NNs have learning capabilities that make them suitable for adaptive control. They can adapt to changes in the environment or system conditions, allowing the controller to continuously improve its performance.

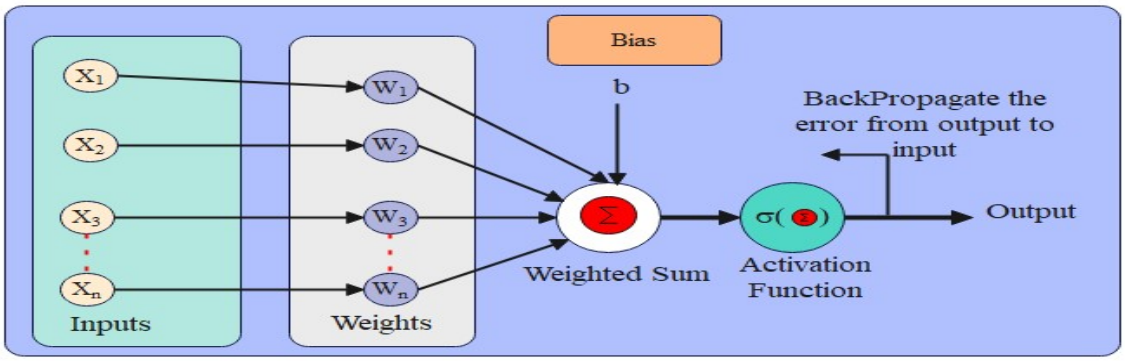


Figure 4.13: Backpropagation of errors(Computation Algorithm)

Effective for time-invariant systems: NN controllers have been found to be particularly useful for time-invariant systems, where the dynamics of the system remain constant over time.

Parameter adjustment based on observed errors: To design an NN-based controller, its parameters need to be adjusted based on observed errors. By propagating these errors through the NN structure and using backpropagation algorithms, the controller parameters can be updated to improve performance.

Parallel computation and real-time implementation: NNs consist of massive parallel computation structures, making them well-suited for high-speed calculations. This enables real-time implementation of NN controllers, where fast responses are required.

Fault tolerance: NNs can provide significant fault tolerance. Even if a few weights or connections within the NN are damaged, the overall performance of the controller is not significantly impaired. This makes NN controllers robust against faults or damages.

Overall, NN controllers offer flexibility, adaptability, and robustness, making them a useful option in situations where mathematical models are not available or where adaptive control is needed [46].

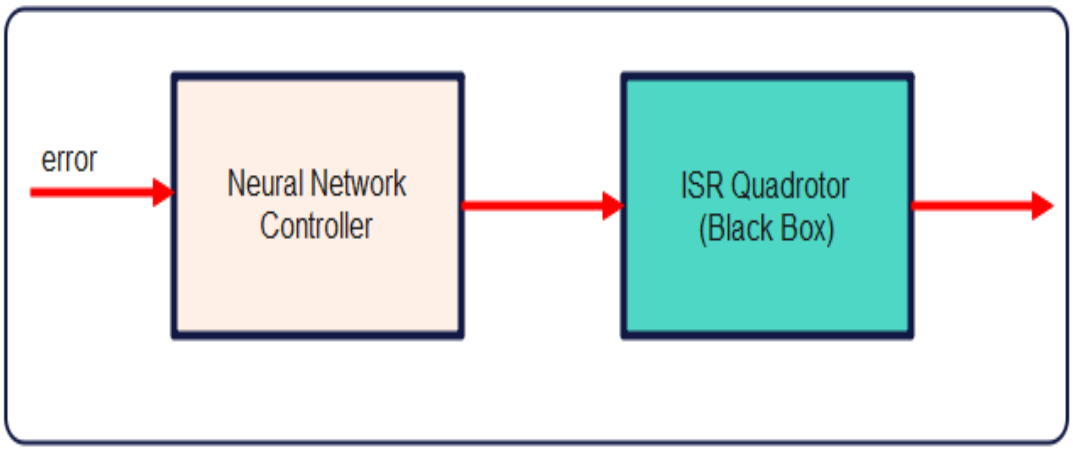


Figure 4.14: NN Control

In this thesis, a Feedforward Neural Network Architecture was implemented. The input-output data from the FPID-GA controller was utilized for training the network. The training process involved using the Levenberg-Marquardt backpropagation algorithm through MATLAB

NN toolbox. The sigmoid function was employed as the activation function for the network. To execute the application and load the input-output data from the workspace, the command "nnstart" was used in the MATLAB command window. The training was performed iteratively until the desired performance level was achieved. Therefore, the Levenberg-Marquardt training algorithm method was applied in this thesis.

4.3.2.3 Network Architecture

As mentioned in the previous section, a Feedforward Network Architecture was employed in this thesis. The network's inputs consisted of two variables: the error and its rate. For the hidden layer, a total of 10 neurons were used, which remained identical across all dynamics. The backpropagation algorithm was employed to train the network, with the sigmoid function serving as the activation function. The weight and bias were adjusted internally through the training algorithm. The architecture of the network for all dynamics can be seen in Figure(4.15)below.

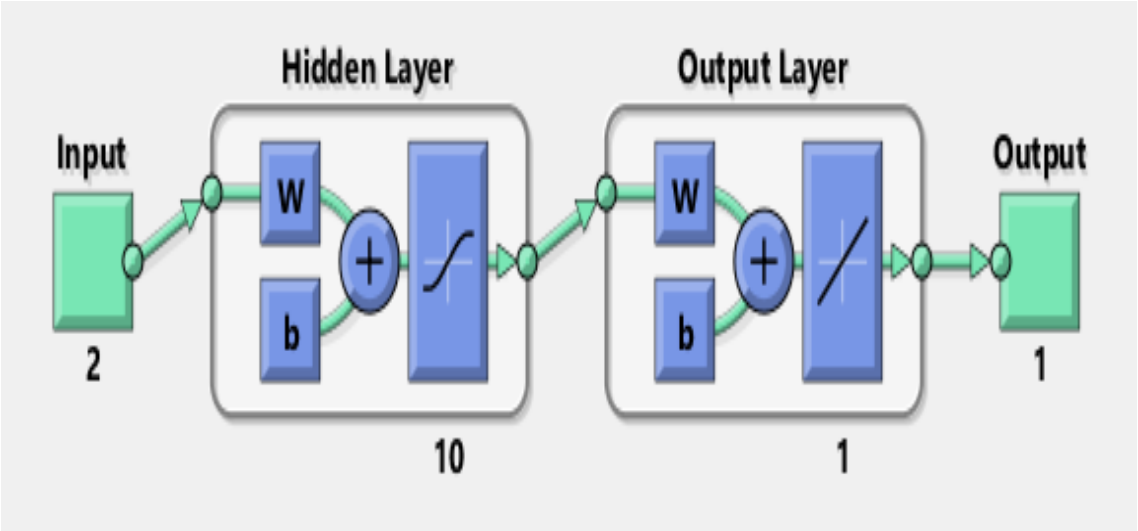


Figure 4.15: The Internal structure of Proposed Controller(NNFPID)

4.4 Fixed point Tracking(Step Response)Control

In this section fixed point tracking of the ISR Quad-rotor control algorithm using GA Optimized Fuzzy-PID Based Neural Network Controller for the desired response was taken as step(x=y=z=1m) all the position and altitude was taken as unity and the heading angle was 45^0. The controller was designed based on this desired values in below sections.

4.5 Position and Altitude controller

In this section the positions and altitude controller was implemented based on input output data from GA Optimized Fuzzy-PID Controller by training using NN Toolbox and the result of the control performances and fitness properties are described in below subsections for each controller.

4.5.1 Position x Controller

Using the data sets of position x controller input output from the GA Optimized Fuzzy-PID Controller and training it the the result of the control performances after 12 iterations and the response was as in Figure(4.16).

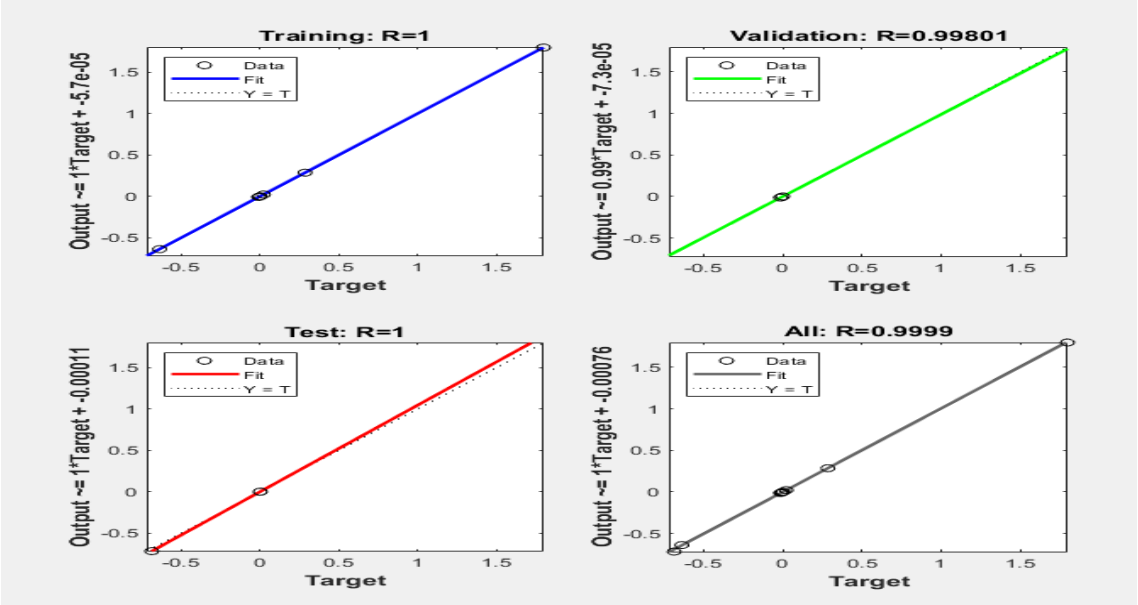


Figure 4.16: Training Result for Position x Controller after 12 Iterations

4.5.2 Position y Controller

Using the data sets of position y controller input output from the GA Optimized Fuzzy-PID Controller and training it the result of the control performances after 109 iterations and the response was as in Figure(4.17).

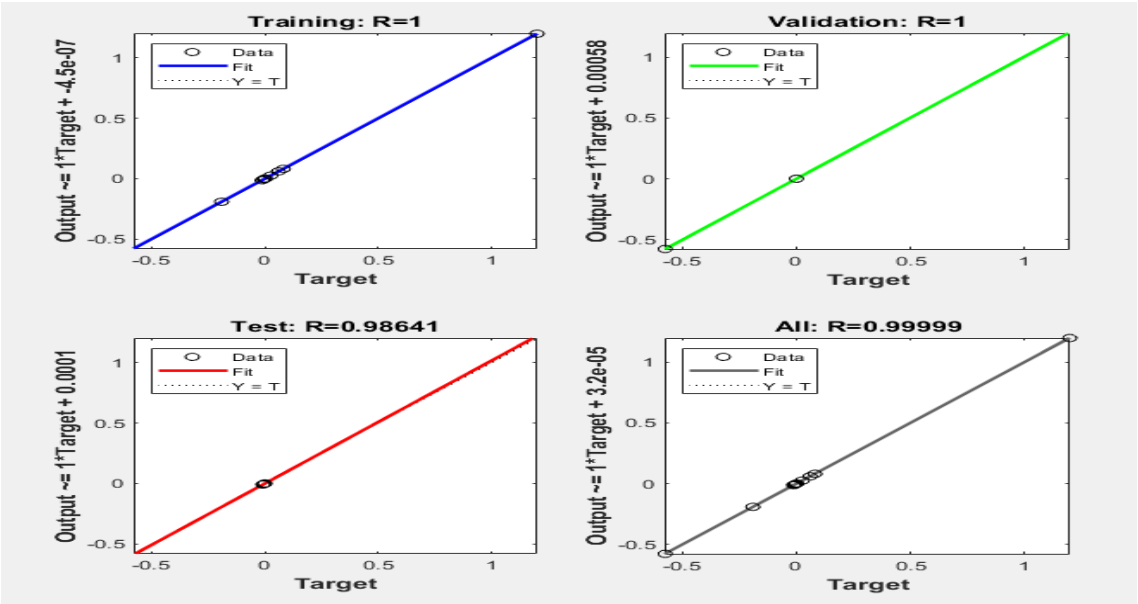


Figure 4.17: Training Result for Position y Controller after 109 Iterations

4.5.3 Altitude z Controller

Using the data sets of altitude z controller input output from the GA Optimized Fuzzy-PID Controller and training it the result of the control performances after 417 iterations and the response was as in Figure(4.18).

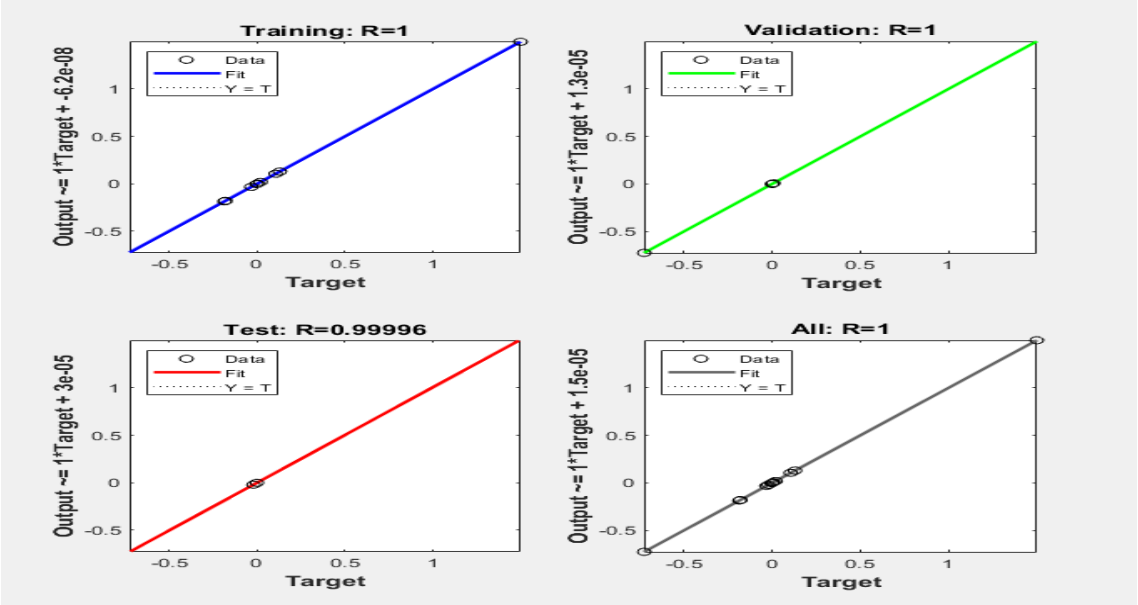


Figure 4.18: Training Result for Altitude z Controller after 417 Iterations

4.6 Attitude and Heading controller

In the previous section above the positions and altitude controller was implemented and discussed the same technique was applied by training using NN tool box and the result of the control performances and fitness properties are described below for each controllers.

4.6.1 Heading Yaw (ψ) controller

Using the data sets of heading angle 45 degree for yaw controller input output from the GA Optimized Fuzzy-PID controller and training it the result of the control performances after 8 iterations and the response was as in Figure(4.19).

4.6.2 Attitude Roll(ϕ)controller

Using the data sets attitude Roll(ϕ)controller input output from the GA Optimized Fuzzy-PID Controller and training it the result of the control performances after 8 iterations and the response was as in Figure(4.20).

4.6.3 Attitude Pitch(θ)controller

Using the data sets attitude Pitch(θ) controller input output from GA Optimized Fuzzy-PID Controller and training it the result of the control performances after 9 iterations and the response was as in Figure(4.21).

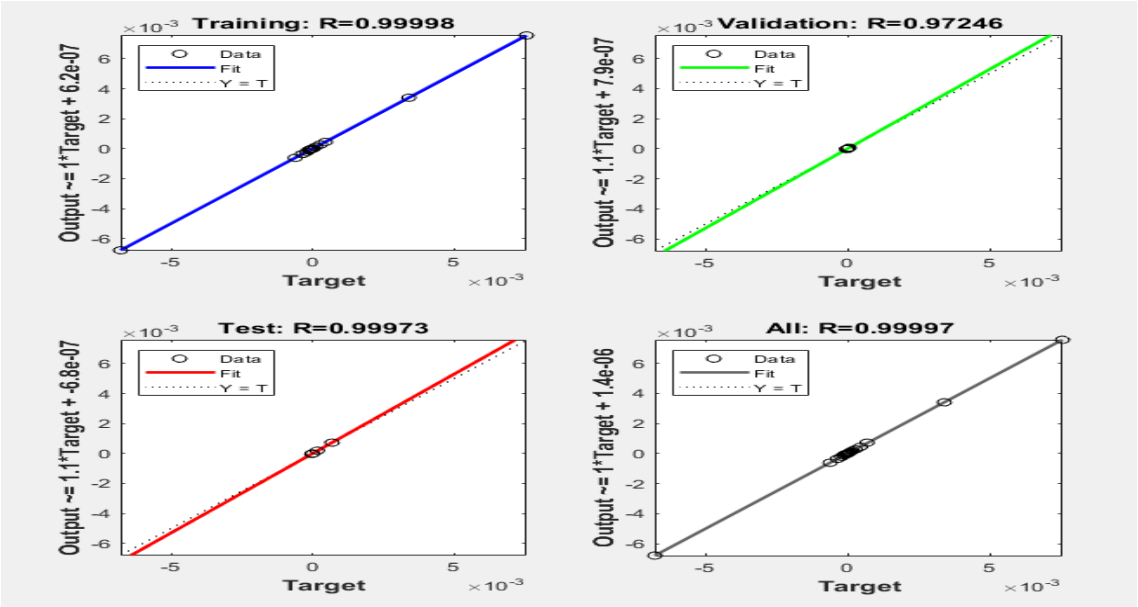


Figure 4.19: Training Result for Heading(ψ) Controller after 8 Iterations

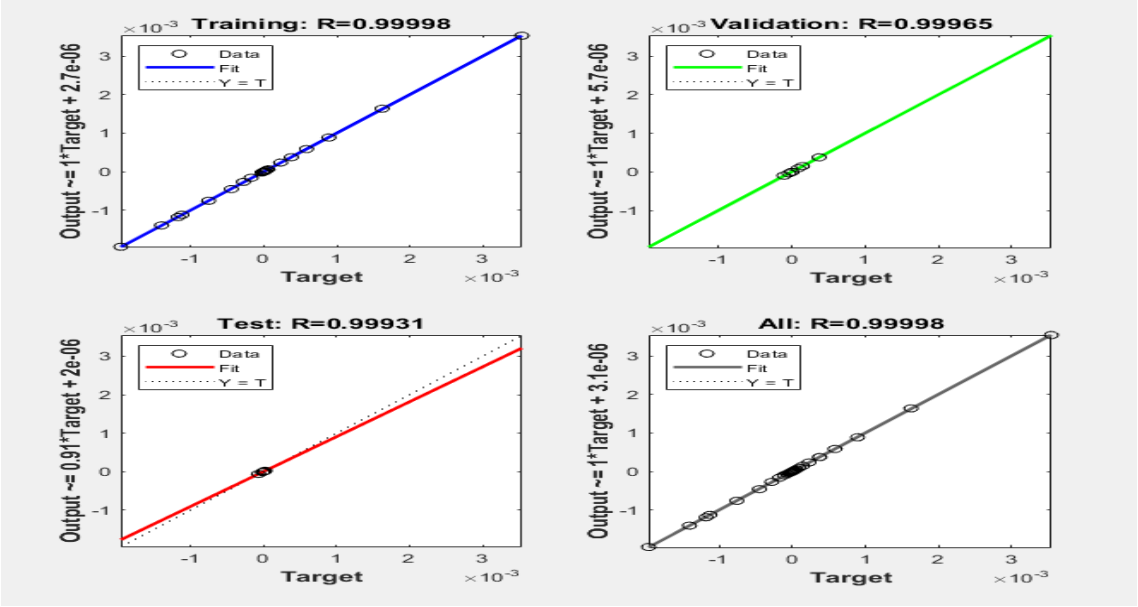


Figure 4.20: Training Result for Roll(ϕ)Controller after 8 Iterations

4.7 Trajectory Tracking Control

In the previous sections, a Fixed Point Tracking Controller was developed for the six dynamics of the ISR Quad-rotor using aGA Optimized Fuzzy-PID Based Neural Network Controller. In this section, will focus on designing a Trajectory Tracking Controller for the proposed plant. This involves using time-varying trajectory desired data from the GA Optimized Fuzzy-PID Controller and generating input-output data from it. This data will then be used to design the GA Optimized Fuzzy-PID Based Neural Network Controller, which serves as the proposed controller for this thesis. The method for designing the GA Optimized Fuzzy-PID Based Neural Network Controller remains the same as the step response controller discussed in the part before this one.

This is where the main distinction is that the desired values in trajectory tracking are

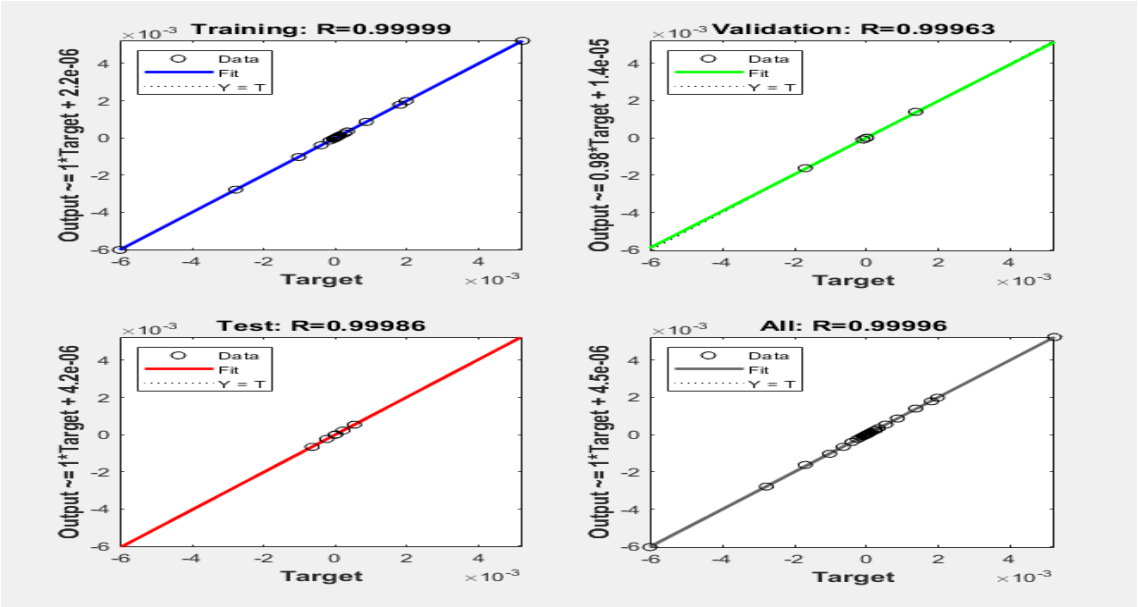


Figure 4.21: Training Result for Pitch(θ) Controller after 9 Iterations

not fixed points but vary with time. This thesis explores more than five trajectory tracking algorithms. The controller design phase follows the same approach as the step response controller, with the only variation being the generation of trajectory data. Some trajectory data is generated based on the application of the proposed drone mission, specifically the ISR (Intelligence, Surveillance, and Reconnaissance) application. The studied trajectory tracking controls include Rectangular, Triangular, Square Wave, Circular, Infinity, and Spiral trajectories. Among these, the first three trajectories are designed based on the mission application of the proposed plant.

4.8 Rectangular Trajectory Tracking Control

In this section of the thesis work, an application-based trajectory tracking controller was designed for a law enforcement Quadrotor (ISR) mission. The controller was designed based on applied rectangular trajectory data. The purpose of the drone is to detect the face of a wanted person (Person of Interest) and continue to track them. The maximum height of a human does not exceed 3m, so for this application-based trajectory in the thesis, an altitude of 5m and a heading angle of 45 degrees for the camera view were used. It is assumed that the speed of the drone is 0.5 m/s.Using the above information, way points (trajectory) were planned in the form of rectangular and triangular patterns for the specific operation scenario described in figure(4.22). The training results of the rectangular NNet were described for each dynamic in this section. The same process was followed for the triangular trajectory. The simulation results will be discussed in a later section in Chapter Six.

4.8.1 Position x Controller

The position x controller input and output from the GA Optimized Fuzzy-PID Controller were obtained using the trajectory data sets. By training this data, control performances were evaluated after 33 iterations, and the resulting response is shown in Figure(4.23).

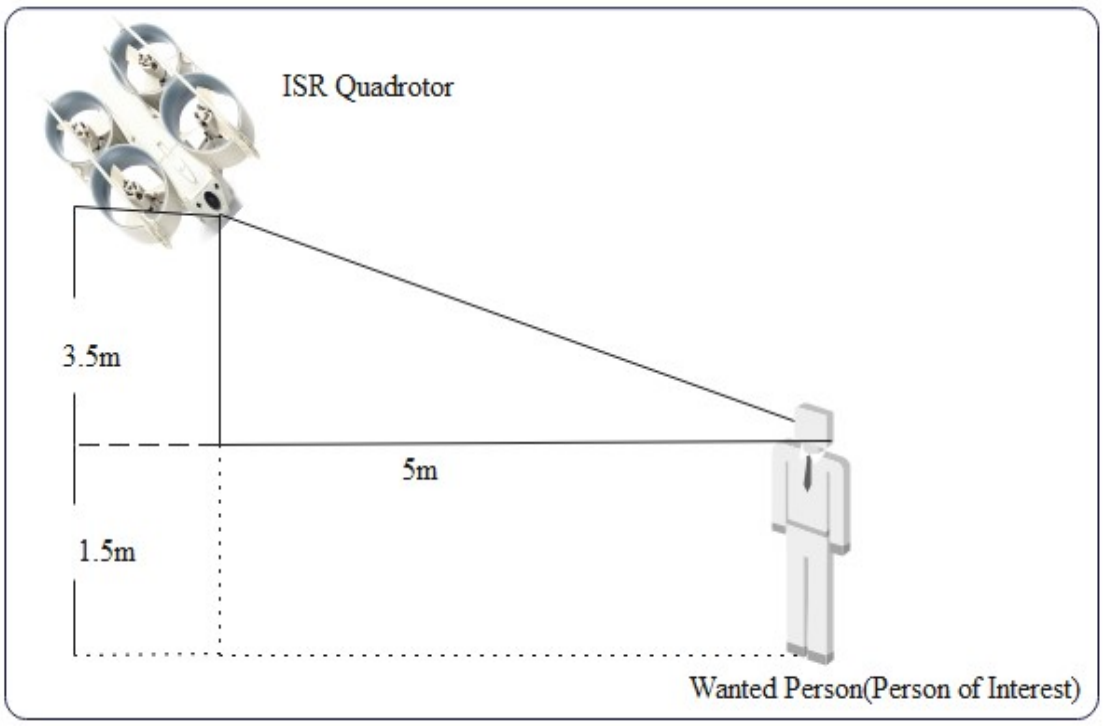


Figure 4.22: Application based Mission Scenario

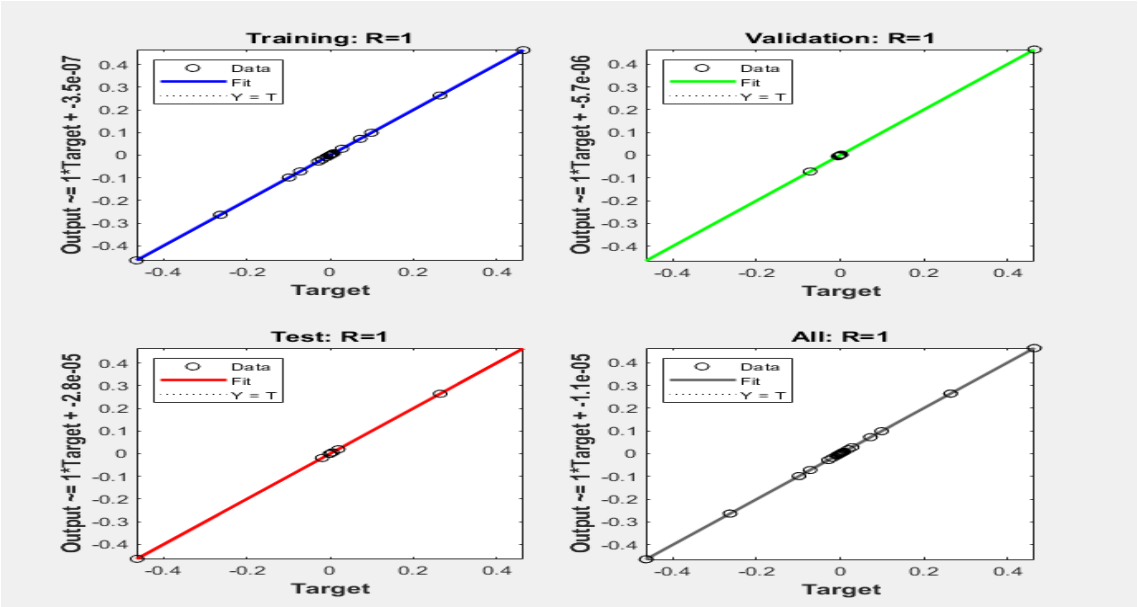


Figure 4.23: Training Result for Position x Controller after 33 Iterations

4.8.2 Position y Controller

The position y controller input and output from the GA Optimized Fuzzy-PID Controller were obtained using the trajectory data sets. By training this data, control performances were evaluated after 36 iterations, and the resulting response is shown in Figure(4.24).

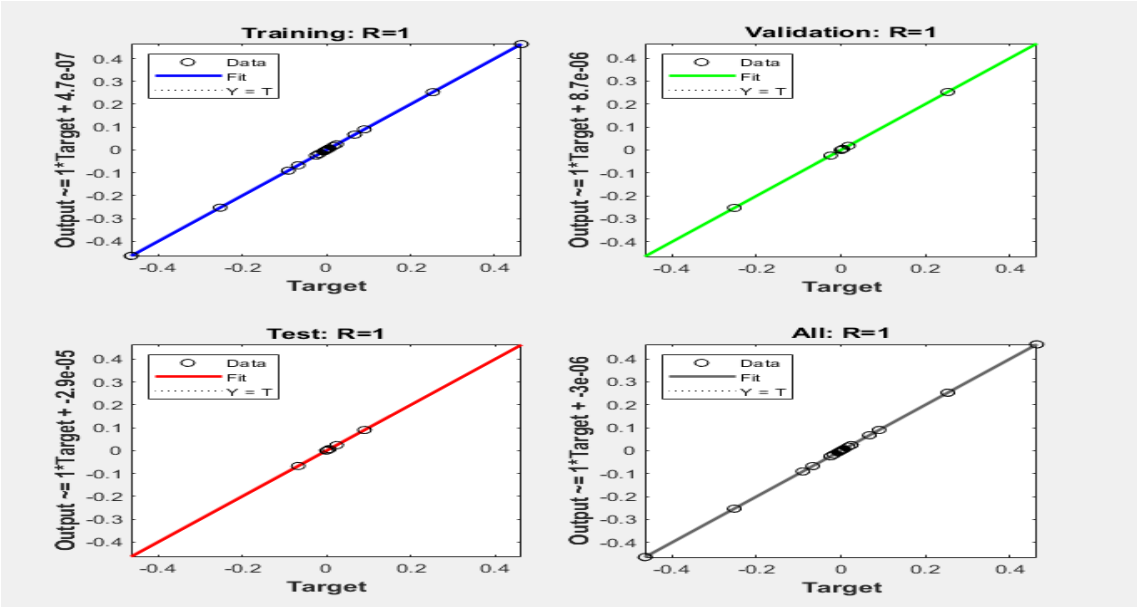


Figure 4.24: Training Result for Position y Controller after36 Iterations

4.8.3 Altitude z Controller

The altitude z controller input and output from the GA Optimized Fuzzy-PID Controller were obtained using the trajectory data sets. By training this data, control performances were evaluated after 10 iterations, and the resulting response is shown in Figure(4.25).

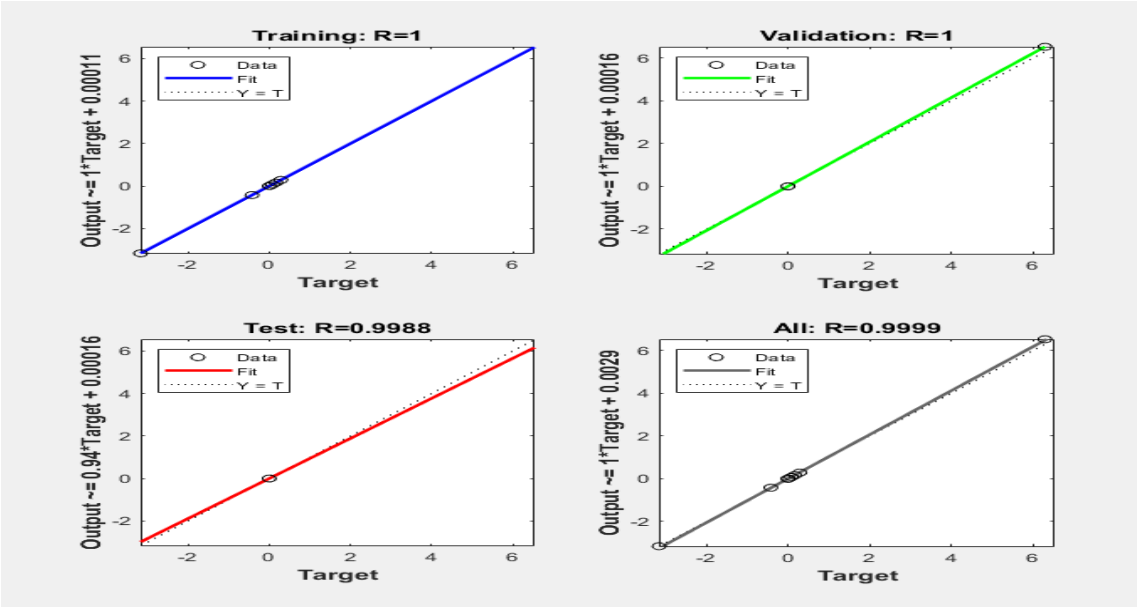


Figure 4.25: Training Result for Altitude z Controller after 10 Iterations

4.8.4 Heading Yaw(ψ)Controller

Using the Trajectory data sets Heading Yaw(ψ)controller input output from the GA Optimized Fuzzy-PID Controller and training it the result of the control performances after 11 iterations and the response was as in Figure(4.26).

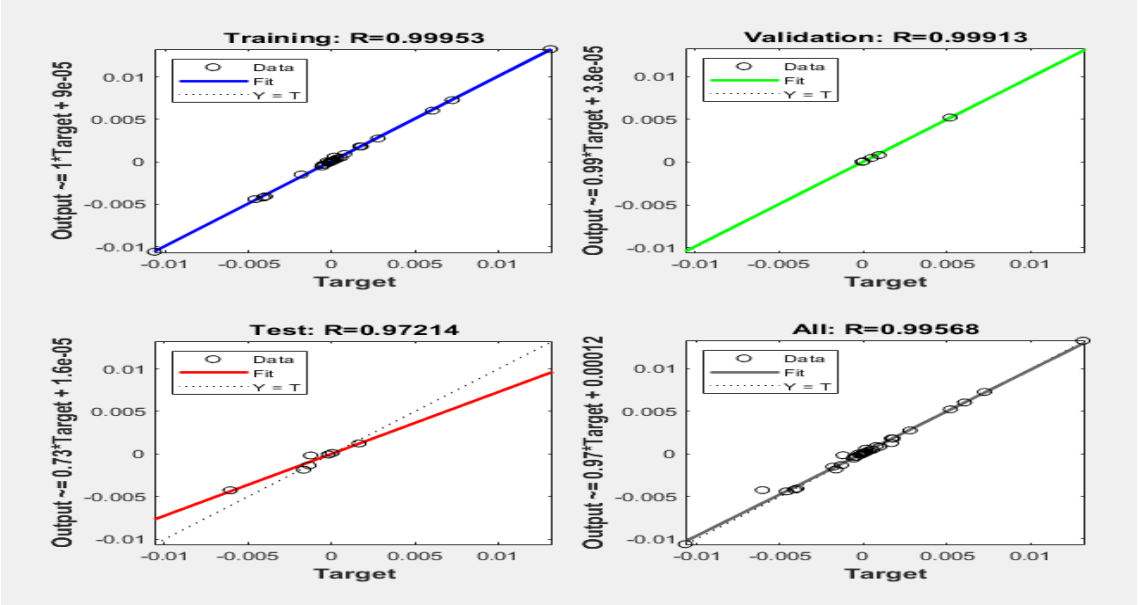


Figure 4.26: Training Result for Heading Yaw(ψ)Controller after 11 Iterations

4.8.5 Attitude Roll(ϕ)controller

Using the Trajectory data sets Attitude Roll(ϕ)controller input output from the GA Optimized Fuzzy-PID Controller and training it the result of the control performances after 9 iterations and the response was as in Figure(4.27).

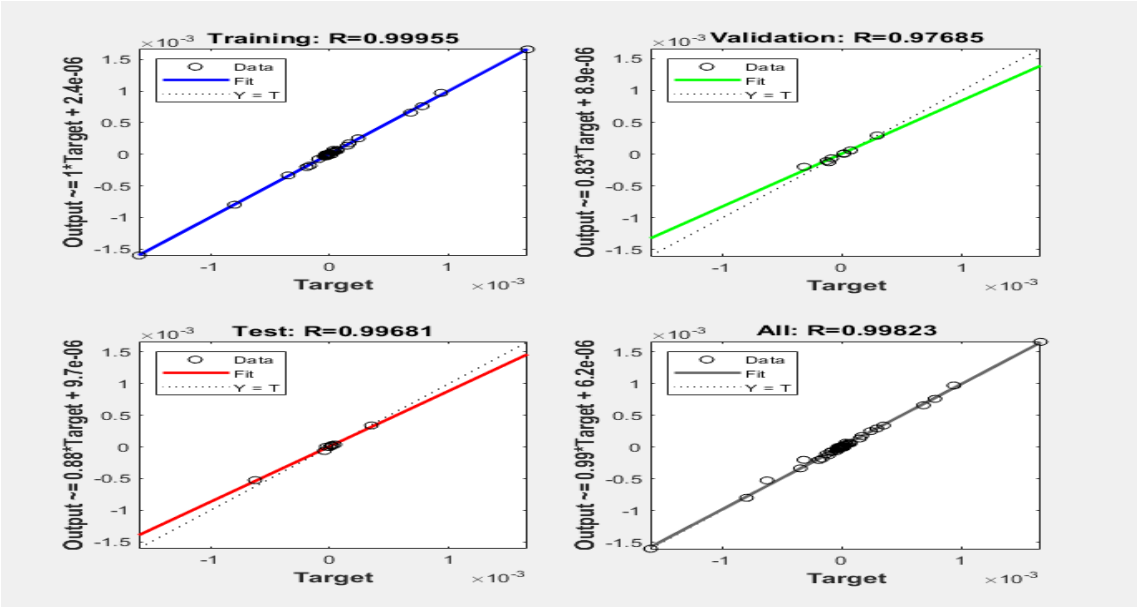


Figure 4.27: Training Result for Attitude Roll(ϕ)Controller after 9 Iterations

4.8.6 Attitude Pitch(θ)Controller

Using the Trajectory data sets Attitude Pitch(θ)controller input output from the GA Optimized Fuzzy-PID Controller and training it the result of the control performances after 7 iterations and the response was as in Figure(4.28).

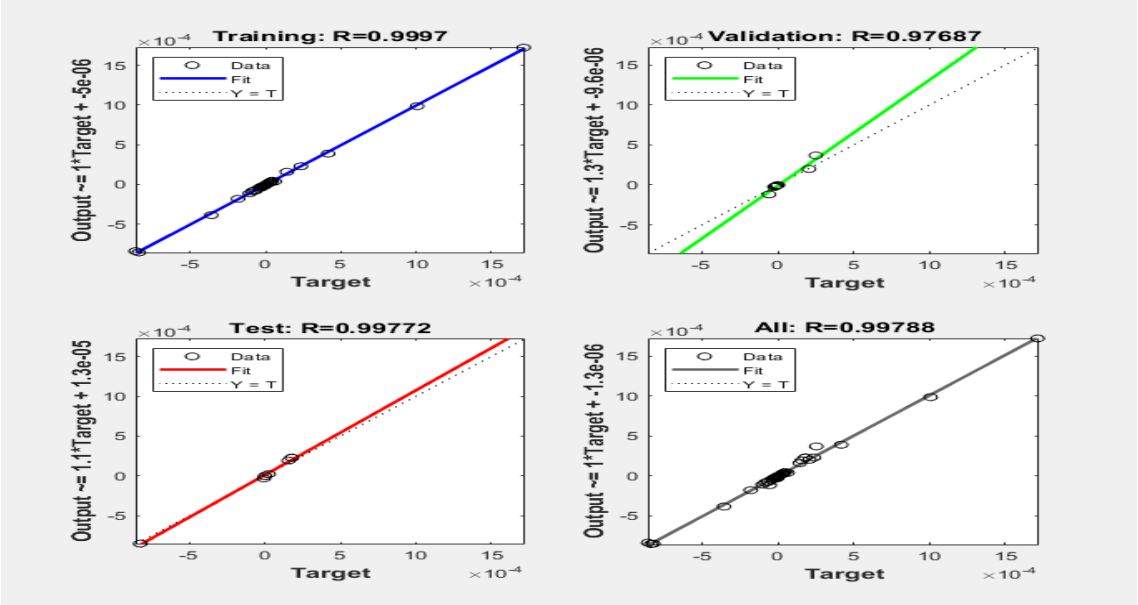


Figure 4.28: Training Result for Attitude Pitch(θ)Controller after 7 Iterations

4.9 Square Wave Trajectory Tracking Control

In this section, a similar approach was followed for the design of the trajectory tracking controller for both Rectangular and Triangular trajectories. The only difference lies in the pseudo code used for generating the trajectory data. In this case, polynomial trajectory data was generated and the training process remained the same. The pseudo code for generating the trajectory can be found in Appendix B.The controller data was fitted with 11 iterations for the x position, 8 iterations for the y position, and 44 iterations for the Altitude z controller. The performance of the controller was excellent Regarding following the desired reference trajectory. This performance will be discussed further in the simulation results and discussion in Chapter Six.

4.10 Infinity Trajectory Tracking Control

For infinity Trajectory tracking the reference signal was taken as for $x = 1 + sin(2 * t)$,$y = 1 - cos(t)$ and at altitude of 2m.The training result was fit good at x position after 12 iteration's,for y position at 481 iterations and for altitude z at 50 iterations.Hence the controller performance was so good tracking of the desired reference trajectory and was discussed in later section in simulation results and discussion in chapter six.

4.11 Circular Trajectory Tracking Control

For circular Trajectory Tracking the reference signal was taken as for $x = 2 * cos(0.314 * t) - 1$,$y = 2 * sin(0.314 * t)$ and $z = cos(0.314 * t)$.The training result was fit good at x position after 7 iteration's,for y position at 217 iterations and for altitude z at 17 iterations.Hence the controller performance was so good tracking of the desired reference trajectory and was discussed in later section in simulation results and discussion in chapter six.

4.12 Spiral Trajectory Tracking Control

For spiral Trajectory tracking the reference signal was taken as for $x = cos(1.05 * t)$,$y = sin(1.05 * t)$ and $z = 3 + (t/4)$.The training result was fit good at x position after 39 iteration's,for y position at 273 iterations and for altitude z at 13 iterations.Therefor the controller performance was so good tracking of the desired reference trajectory and was discussed in later section in simulation results and discussion in chapter six.

Chapter 5

Face Recognition and Tracking with ISR Quad-Rotor

5.1 Introduction

This Chapter focuses on the utilization of ISR quad-rotor for face recognition and tracking. Quad-rotor drones can be operated remotely and programmed to carry out specific tasks as per user instructions. This makes them an ideal choice for monitoring remote areas and densely populated locations without causing any harm to operators. The use of drones can aid law enforcement agencies in locating wanted individuals or person of interest in crowded places like stadiums, festivals, and inaccessible regions during disaster relief operations. In Ethiopia, there are numerous religious and cultural festivals where law enforcement agencies commonly rely on physical security measures. However, this thesis aims to explore the implementation of drone-assisted face recognition and tracking technology to enhance public safety and maintain peace and security in such events. By targeting wanted individuals using drone technology, law enforcement agencies can effectively protect the public and uphold the rule of law. The next section of this chapter will focus on the implementation of face recognition/identification using Python, as well as discuss operational scenarios involving semi-autonomous and autonomous face tracking algorithms.

5.2 Face Recognition with Python

Face recognition is a popular computer vision technique that involves identifying or verifying a person's identity based on their facial features. Python provides several libraries and frameworks that make face recognition implementation easier. Here's a step-by-step guide on performing face recognition with Python: **Install Required Libraries:**

'pip install opencv-python' to access OpenCV dlib: type 'pip install dlib facial recognition Installing face recognition on Pip**Pre pare Dataset:** Collect images of the individuals who wanted to be recognize and create a dataset. Ensure that each person has a sufficient number of images taken from different angles and under various lighting conditions.**Face Encoding:** Use the face recognition library to encode the faces in dataset. Load each image and generate a unique face encoding vector for each face using the 'face recognition.face encodings()' function.**Face Recognition:** Capture or load an image containing faces to be recognized. Use the 'face recognition.face locations()' function to detect the face locations

in the image. Generate face encodings for the detected faces using 'face recognition.face encodings()'. Compare these face encodings with the encodings in dataset using the 'face recognition.compare faces()' function. Can iterate through the detected faces and check for similarities to determine the identity of the person. There are different methods and libraries available in Python for implementing face recognition. Two popular approaches are: **Face Recognition Library:** The face recognition library is a powerful and user-friendly Python library that provides pre-trained models for face recognition tasks. It is built on top of dlib, a C++ library known for its effectiveness in face detection and facial landmark identification. The face recognition library allows to perform face recognition tasks such as face detection, face identification, and face comparison with ease [47].

OpenCV (Open Source Computer Vision Library): OpenCV is a robust computer vision library that also provides face recognition functionality. It offers ready-to-use algorithms and functions for facial detection, tracking, and recognition. OpenCV is highly optimized for efficiency, making it suitable for real-time applications such as video surveillance or facial authentication.

When implementing face recognition with Python, can choose the approach that best suits requirements and familiarity with the respective libraries. Both options provide efficient and accurate face recognition capabilities [47].

Overall, Python offers a range of tools and libraries to facilitate face recognition tasks, making it a popular choice for developers and researchers in the field.

5.2.1 Open CV

OpenCV (Open Source Computer Vision Library) is indeed one of the most popular and powerful libraries for computer vision tasks. It was originally written in C/C++, but it now provides convenient bindings for Python as well [48].

One of the key features of OpenCV is its ability to utilize machine learning algorithms for various tasks, including face detection. Detecting faces within an image can be challenging due to the complexity and variability of facial expressions, poses, and lighting conditions. OpenCV tackles this problem by employing machine learning-based classifiers [48].

These classifiers consist of thousands of smaller patterns and features that are used to identify faces. Instead of trying to identify a face all at once, the algorithms decompose the task into numerous smaller and simpler tasks. Each of these smaller tasks is designed to solve a specific aspect of face recognition, making the overall task more manageable.

Moreover, OpenCV is highly optimized for computational efficiency and real-time applicat ions. This makes it an ideal choice for implementing real-time face recognition systems that utilize a camera feed, such as in video surveillance or biometric authentication applications [49].

Overall, OpenCV combines the power of machine learning algorithms with its focus on real-time performance, making it a versatile and widely used library for a variety of computer vision tasks, including face detection and recognition [49]the Phyton code for face Identification/Recognition was at Appendix C and builtin function were used for face recognition [49].

According to the training results shown in Figure (5.3), the accuracy of identifying the person of interest is not below 95.61%. This lower accuracy was observed even when the wanted person did not open their eyes, as seen in the training result figure. In the

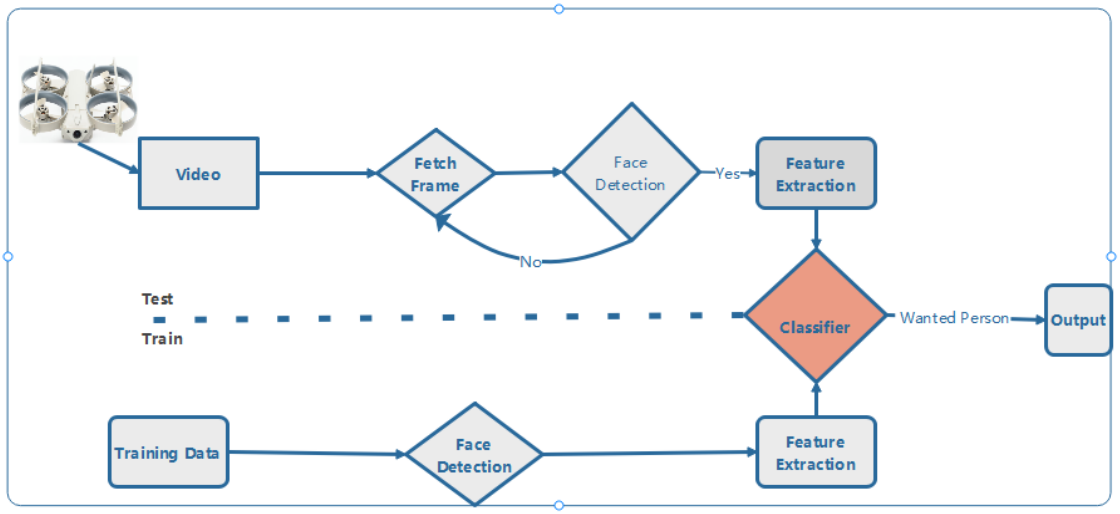


Figure 5.1: Flow Chart for Face Recognition and Identification Phases

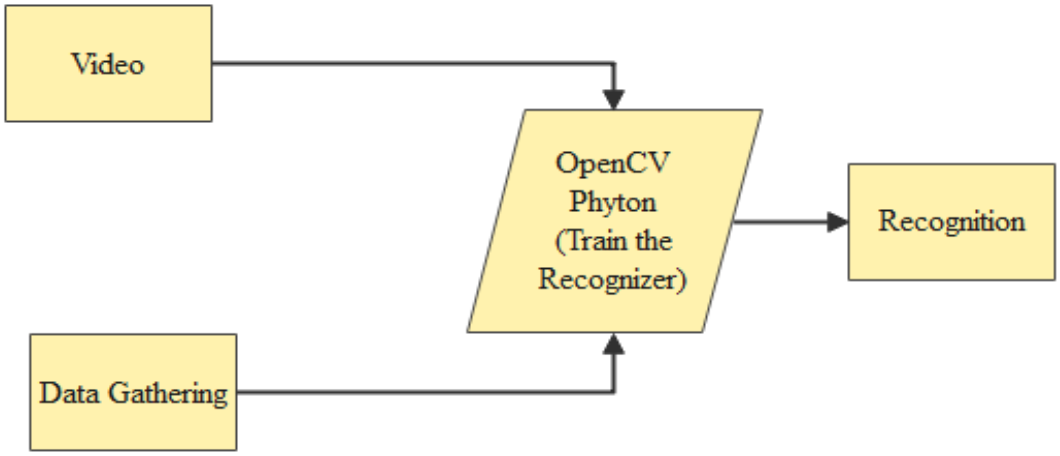


Figure 5.2: Face Recognition and Identification Phases

normal face position, the accuracy of identifying the person of interest was 99.17%. In other face conditions, the accuracy of recognizing the wanted person was 98.65%, as depicted in Figure(5.4).

5.3 Face Recognition and Tracking Algorithm

Case 1:Operational Scenario(semi Autonomy)

This thesis focuses on developing a real-time face recognition drone surveillance system. The system involves flying a drone at a low height, less than 5 meters above the ground, equipped with a webcam camera for capturing the camera feed. A laptop serves as the server, running a Python program that continuously scans all faces in view of the drone's camera.

The system compares the detected faces against a pre-inserted dataset of criminals' faces, allowing it to identify any matches. The face detection feature is installed on the server laptop, which can be connected through WiFi, enabling the system to run on devices with

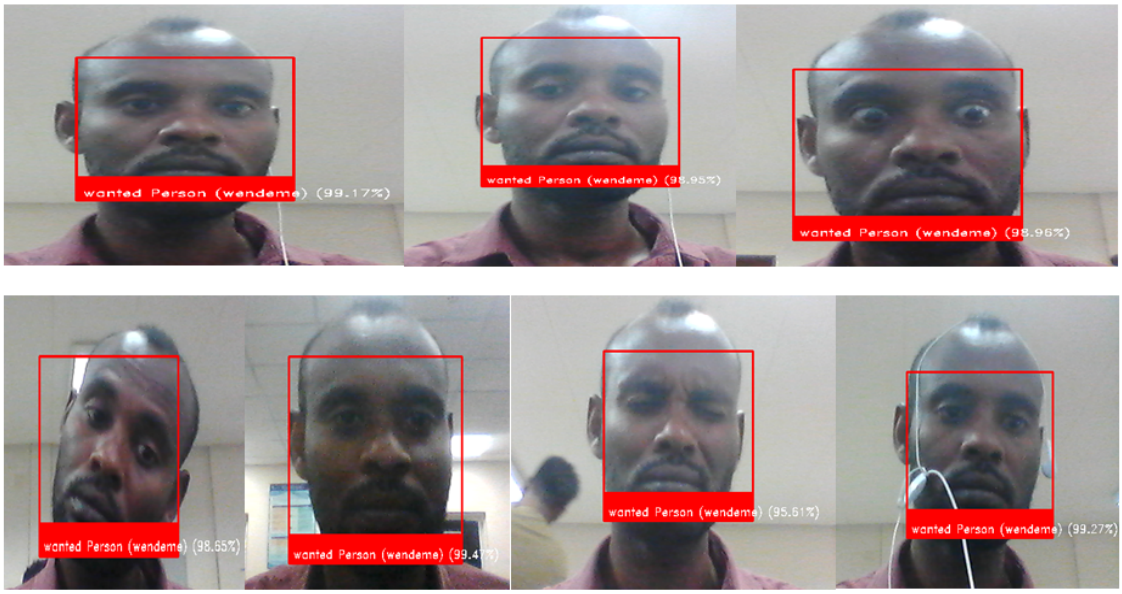


Figure 5.3: Training Result at different Face Conditions

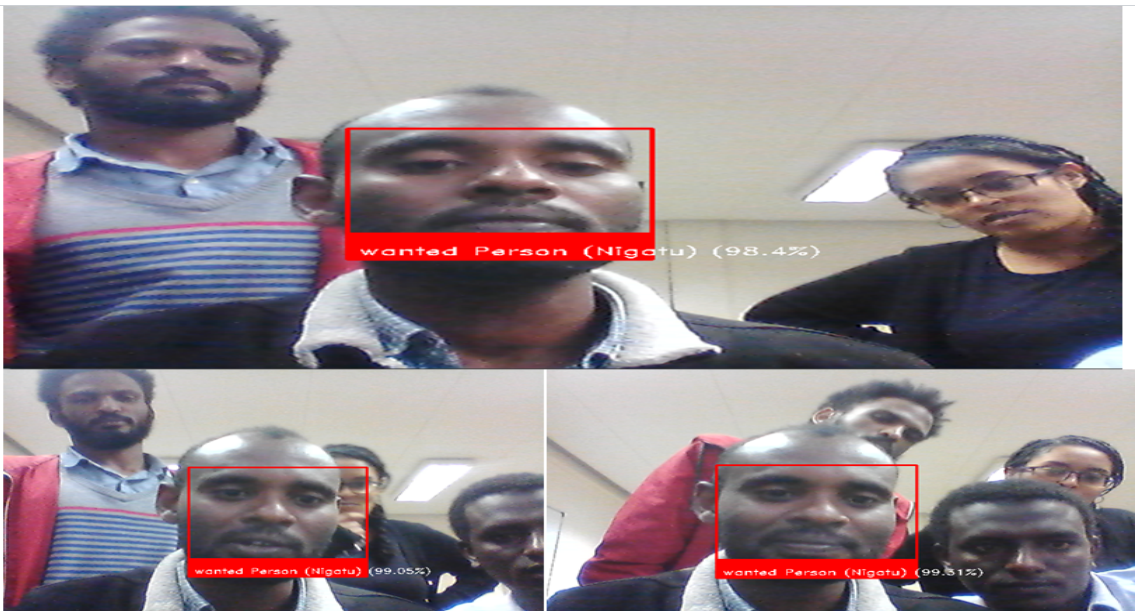


Figure 5.4: Training accuracy Identification/Recognition of Wanted Person (Person of interest) from Others

limited computational power, such as laptops or smartphones.The drone is programmed to fly semi-autonomously within a specified area for demonstration purposes, but it can also follow a programmed flight plan when instructed to track a recognized face. The operational crew consists of a surveillance room controller, responsible for controlling the face recognition and identification system on the Python program, and a drone operator and assistant, in charge of operating the drone in public places like religious events, stadiums, riots, and cultural festivities.

The system also involves security officers who assist in tracking criminals based on the commands given by the surveillance room operator and drone operators at remote locations. Communication between the surveillance room operator, field drone operators, and other security officers is facilitated through military radios. WiFi is utilized for communication

between the surveillance room and the drone camera, allowing the transmission of real-time video feed using IP addresses.

This thesis project has the potential to incorporate real-time face recognition, person detection from video, and the detection of suspicious activities. The role of the drone is solely to capture and transmit real-time video to the laptop server in the surveillance room. It may receive commands to adjust its flight movements smoothly as directed by the operators. The operational scenario is depicted in Figure (5.5).

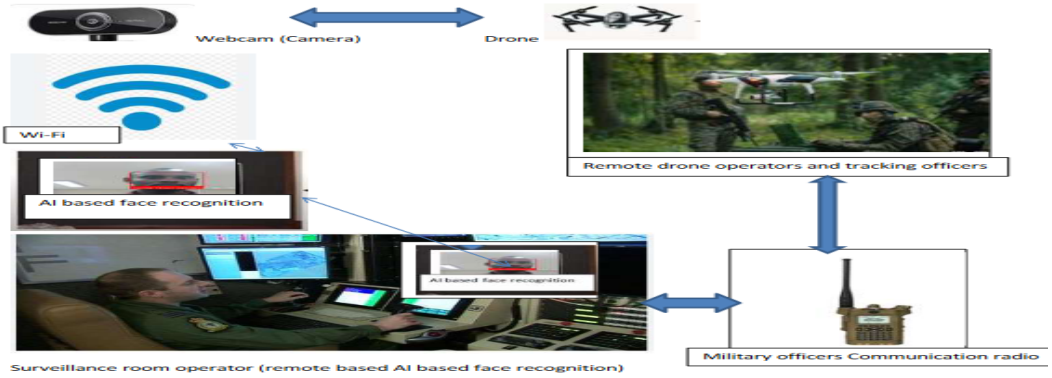


Figure 5.5: Complete ISR Mission using Quad-Rotor (Operational Scenario)

Case 2:Autonomous Face Recognition and Tracking by ISR Quad-Rotor

In the previous section, discussed face identification and tracking that was done semi-autonomously. However, in this section, Will focus on autonomous face recognition and tracking. This means that the altitude of the Quadrotor remains fixed, and the positions of the face of the person of interest (wanted person) are fed to the Quadrotor's trajectory through the camera. The drone then follows the wanted person, and law enforcement professionals enforce the rule of law during the drone's pursuit. To complete the project, a Python-based face recognition system was integrated with MATLAB® Simulink. The ISR Quadrotor Dynamics trajectory received the reference positions based on the wanted person's face positions. These reference positions were used as the trajectory for the drone, enabling it to follow the camera, which in turn tracks the wanted person. Figure (5.6)illustrates the scenario of autonomous face tracking using the ISR Quadrotor UAV.

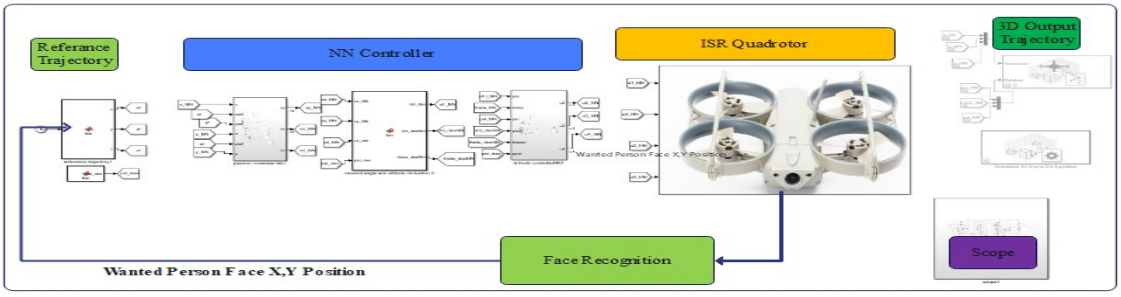


Figure 5.6: Block Diagram for Autonomous Face Recognition and Tracking by ISR Quad-Rotor

Chapter 6

Simulation Results and Discussion

6.1 Introduction

The ISR quad-rotor system, previously modeled mathematically in chapter three, is now available as a Simulink model in this chapter. MATLAB®Simulink is used to simulate the closed-loop control of the system. GA Optimized Fuzzy-PID Controller and GA Optimized Fuzzy-PID Based Neural Network controllers are both used. MATLAB® is used to develop the drone's Simulink model, which takes into account both disturbed and disturbance-free scenarios. The performance of the suggested control system is assessed while the behavior of the ISR quad-rotor model is monitored. The simulation results are shown in comprehensive graphs.

A simulation study is conducted to demonstrate the effectiveness of the proposed controller for an ISR Quadrotor drone. The main objective of the controller is to minimize trajectory error and attain effective trajectory follwing(tracking) and stabilization. A detailed analysis and discussion of the simulation results is presented, emphasizing how well the suggested quadcopter controls the position and attitude of flight dynamics over a range of trajectories.

To validate the efficiency of the proposed controller, simulation results are presented in the form of detailed graphs, offering a comprehensive analysis. Different trajectories, such as rectangular, triangular, square wave, circular, helical, and infinity trajectories, are considered to assess the effectiveness of the proposed control system in trajectory tracking. The performance of the developed neural network (NN) controller for an unmanned aerial vehicle (UAV) is evaluated through flight tests conducted using the UAV Toolbox. These flight tests aim to assess the controller's ability to achieve stable and accurate control in real-world flight conditions. The study showcases the effectiveness and applicability of the UAV Toolbox in validating the performance of advanced control algorithms, such as GA Optimized Fuzzy-PID Based Neural Network controllers. Furthermore, the performance of the proposed controller is compared to that of an GA Optimized Fuzzy-PID controller. Step response and disturbance checking are performed to assess the controller's effectiveness using performance indices. A comparative analysis is conducted for both controllers Concerning trajectory following, considering the effects of disturbances and parameter variations. The results are discussed and interpreted based on performance metrics.

6.2 Rectangular Trajectory Tracking

In the previous section of Chapter Four, the controllers were designed using speed information obtained from the ISR Quad-rotor plant. Additionally, a pseudo code for the rectangular trajectory was generated based on the application mission of the plant. Therefore, in this section, the performance of the controllers for this trajectory was examined, and the results were presented in the form of graph plots shown in Figures(6.1),(6.2),and(6.3).

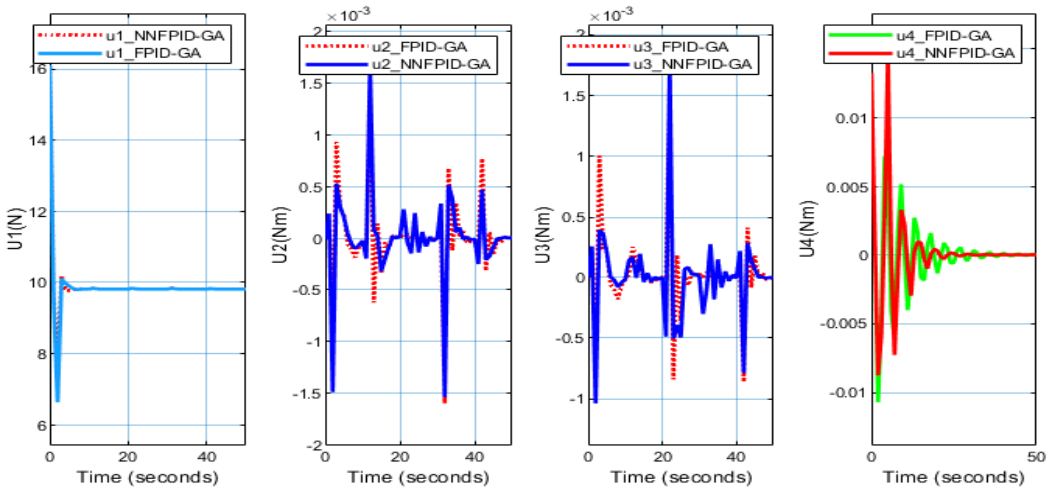


Figure 6.1: Control efforts for Rectangular Trajectory Tracking

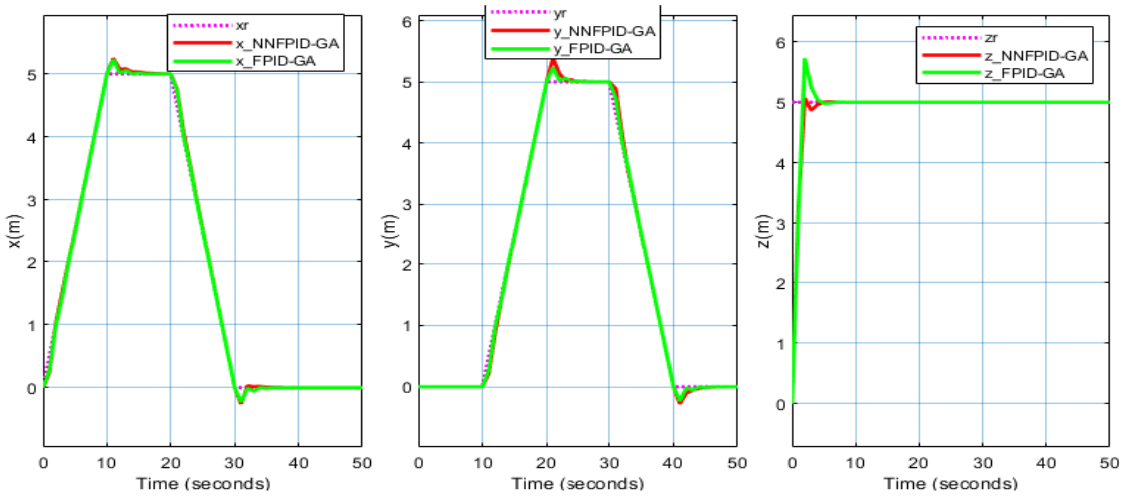


Figure 6.2: Position Tracking Response for Rectangular Trajectory Tracking

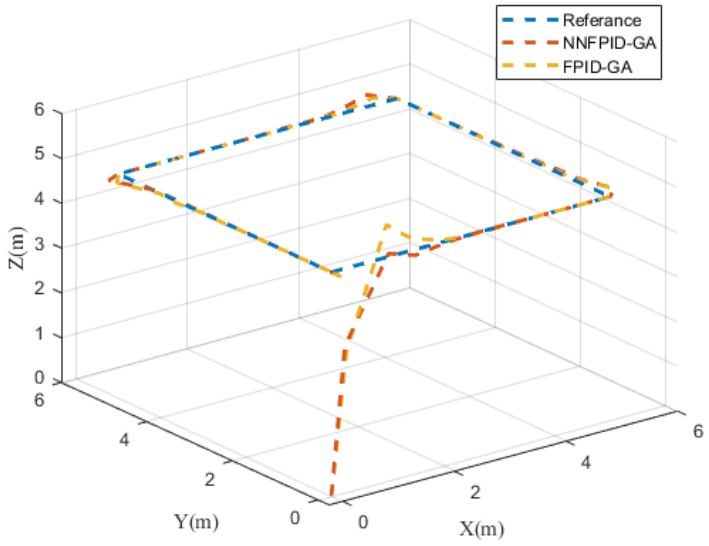


Figure 6.3: 3D Plot for Rectangular Trajectory Tracking

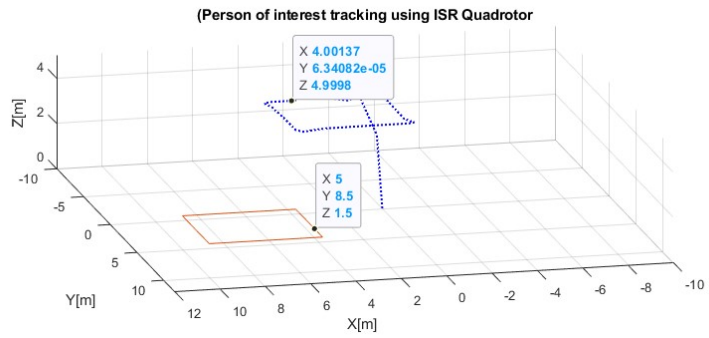


Figure 6.4: Person of interest tracking using ISR Quadrotor

6.3 Triangular Trajectory Tracking

The proposed and GA Optimized Fuzzy-PID controllers were designed in the previous section, Chapter Four, using the speed information of the ISR Quad-rotor. Additionally, a pseudo code for the triangular trajectory was created to fulfill the plant's intended purpose. In this section, the controllers were evaluated for their performance in executing this trajectory. The results are presented in the form of graphs, shown in Figures(6.5),(6.6),and(6.7).

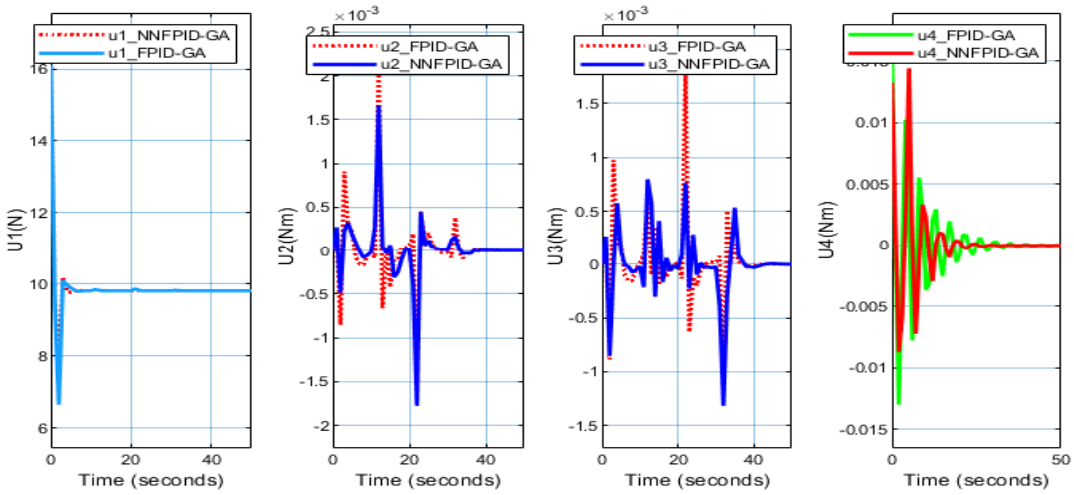


Figure 6.5: Control efforts for Triangular Trajectory Tracking

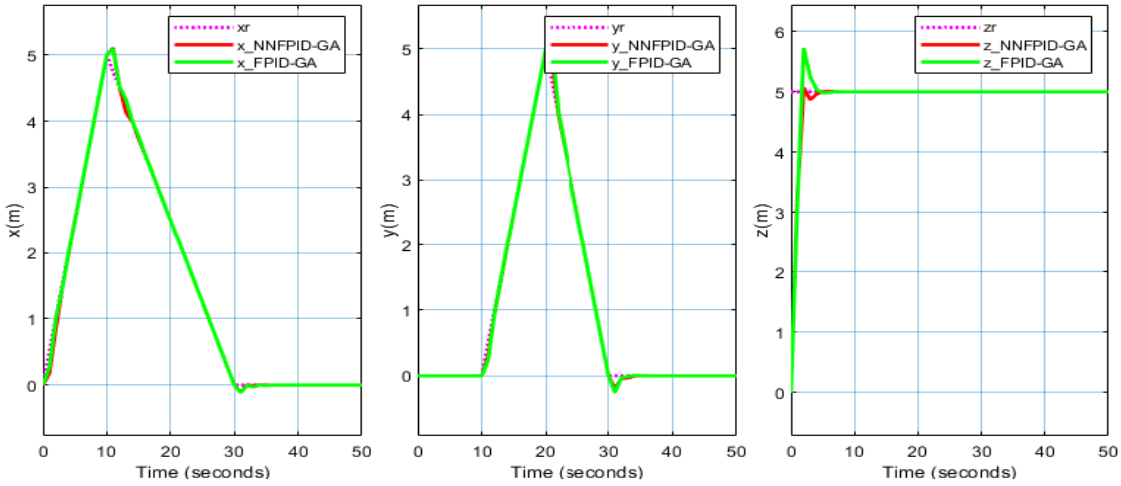


Figure 6.6: Position Tracking Response for Triangular Trajectory Tracking

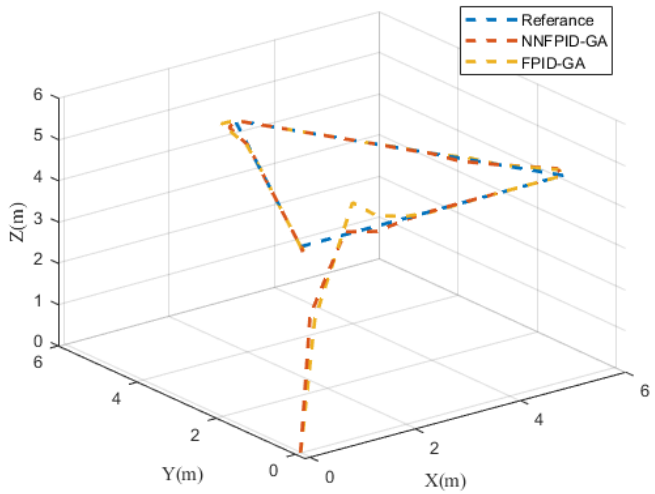


Figure 6.7: 3D Plot for Triangular Trajectory Tracking

6.4 Square Wave Trajectory Tracking

In the previous section of Chapter Four, the controllers were designed using speed information obtained from the ISR Quad-rotor plant. Additionally, a pseudo code for the square wave trajectory was generated based on the application mission of the plant. Therefore, in this section, the performance of the controllers for this trajectory was examined, and the results were presented in the form of graph plots shown in Figures(6.8),(**??**),(6.9),and(6.10).

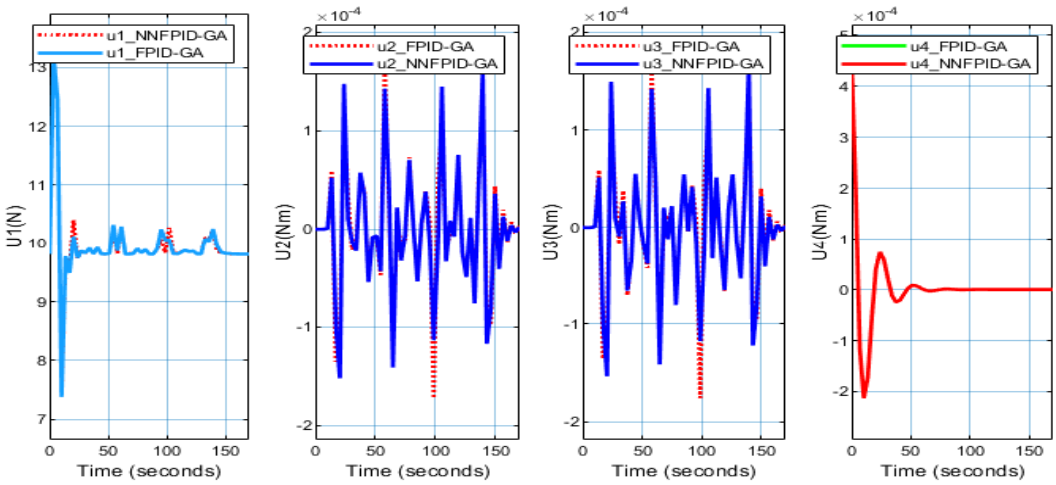


Figure 6.8: Control efforts for Square wave Trajectory Tracking

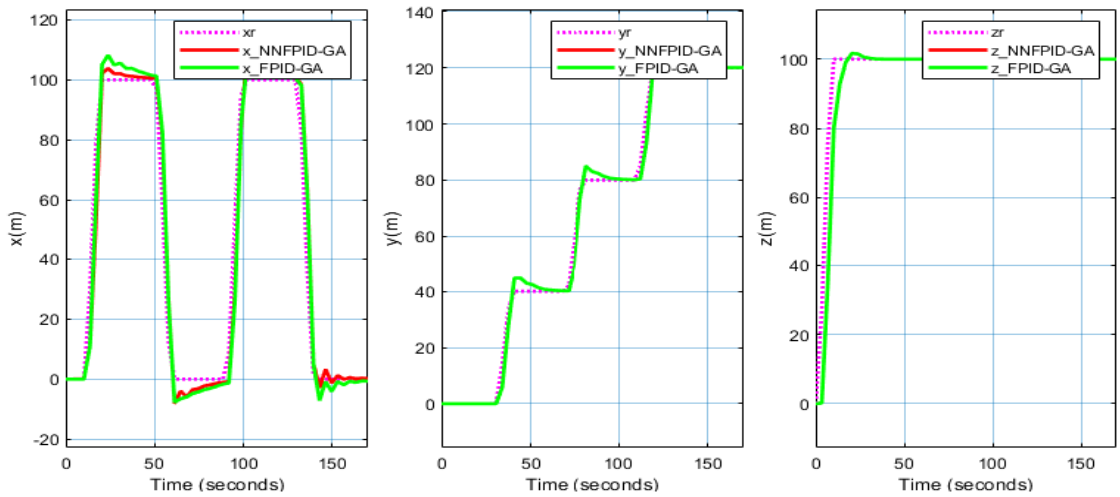


Figure 6.9: Position tracking Response for Square wave Trajectory Tracking

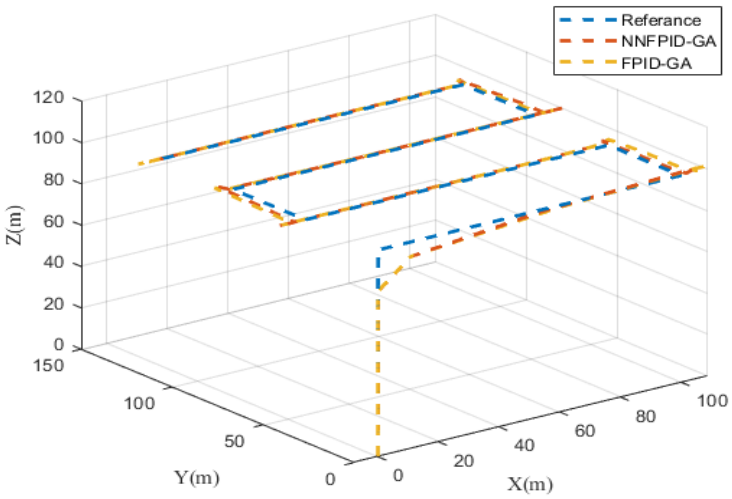


Figure 6.10: 3D Plot for for Square wave Trajectory Tracking

6.5 Circular Trajectory Tracking

In the previous section of Chapter Four, the controllers were designed based on the desired circular trajectory data. Therefore, in this section, the performance of the controllers for this trajectory was analyzed, and the results were presented in the form of graph plots shown in Figures(6.11),(6.12)and(6.13).

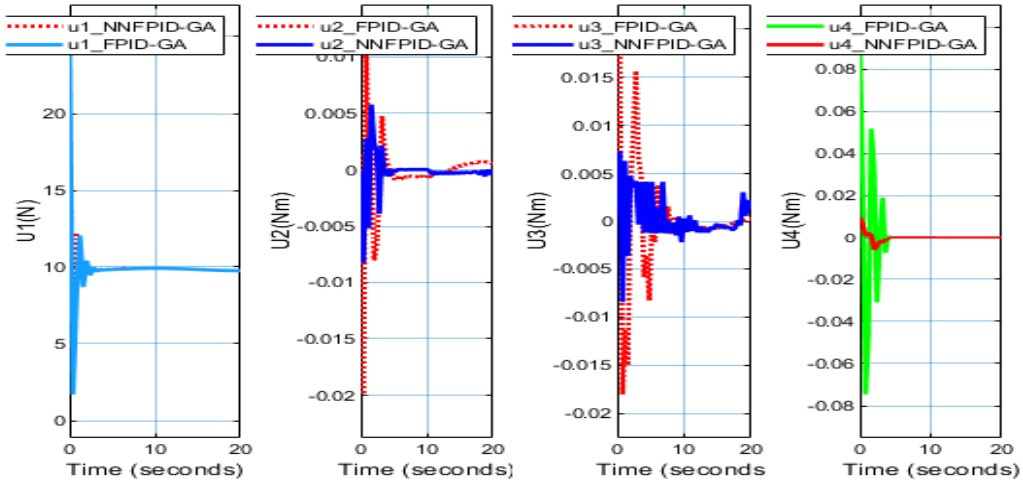


Figure 6.11: Control efforts for Circular Trajectory Tracking

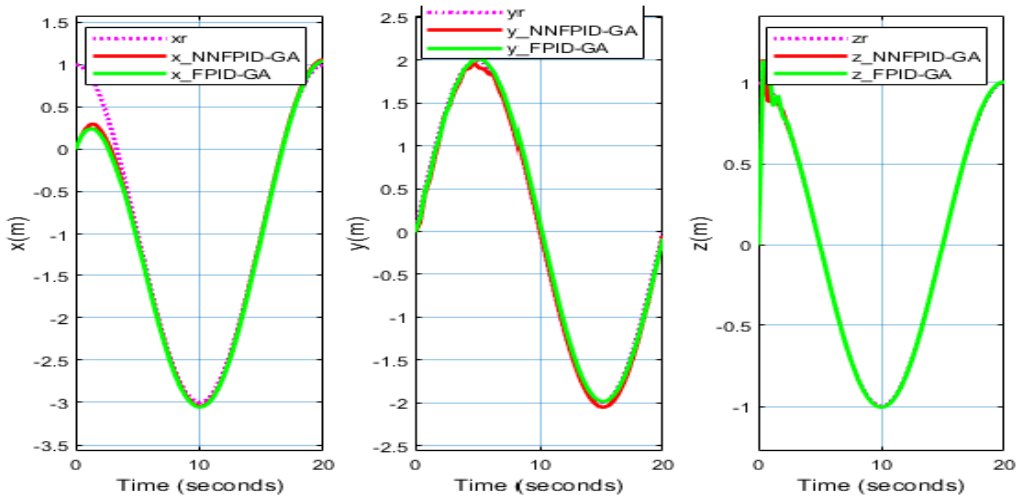


Figure 6.12: Position Tracking Response for Circular Trajectory Tracking

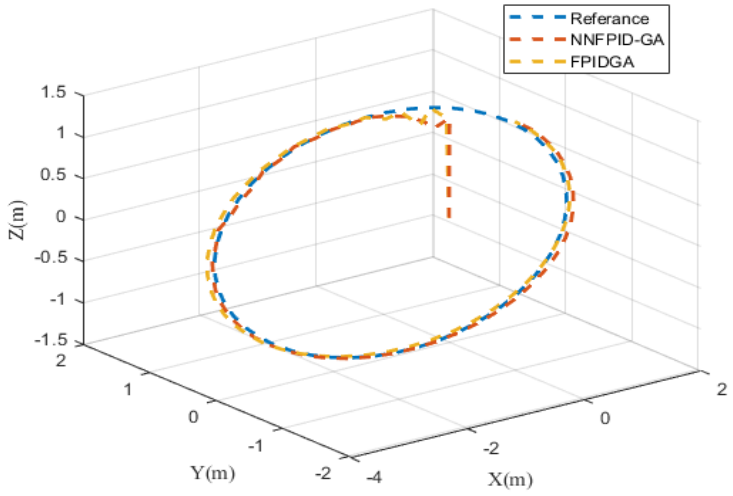


Figure 6.13: 3D Plot for Circular Trajectory Tracking

6.6 Infinity Trajectory Tracking

In the previous section of Chapter Four, the controllers were designed based on the desired infinity trajectory data. Therefore, in this section, the performance of the controllers for this trajectory was analyzed, and the results were presented in the form of graph plots shown in Figures(6.14),(6.15)and(6.16).

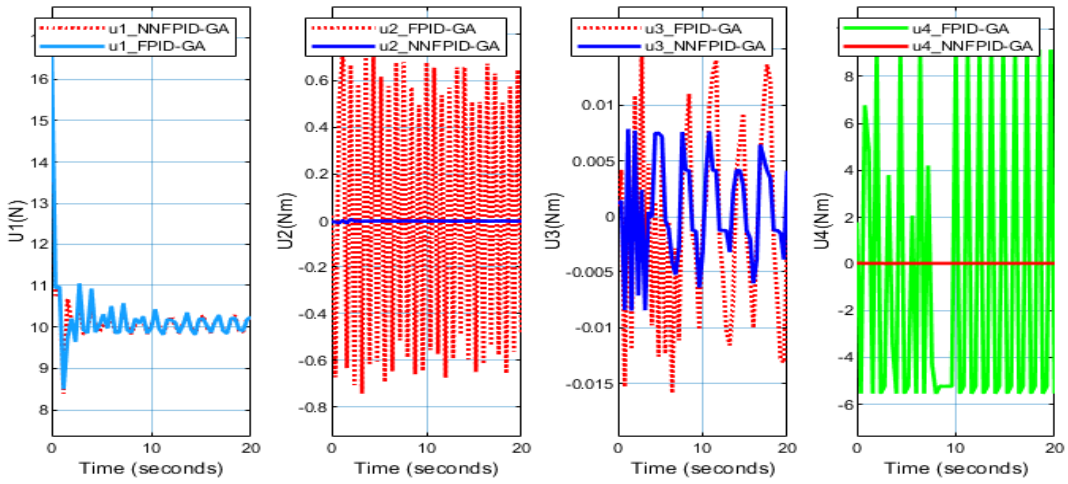


Figure 6.14: Control efforts for Infinity Trajectory Tracking

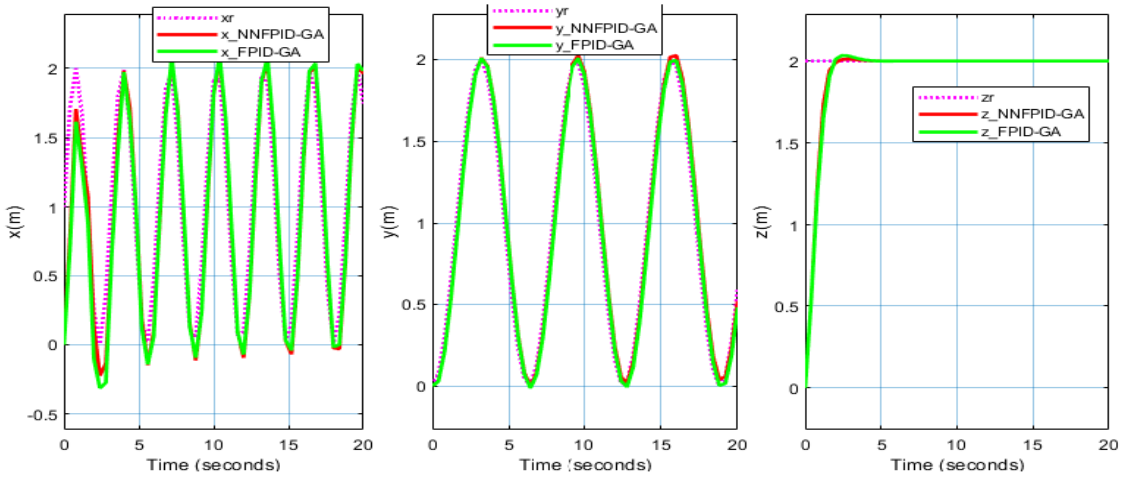


Figure 6.15: Position Tracking Response for Infinity Trajectory Tracking

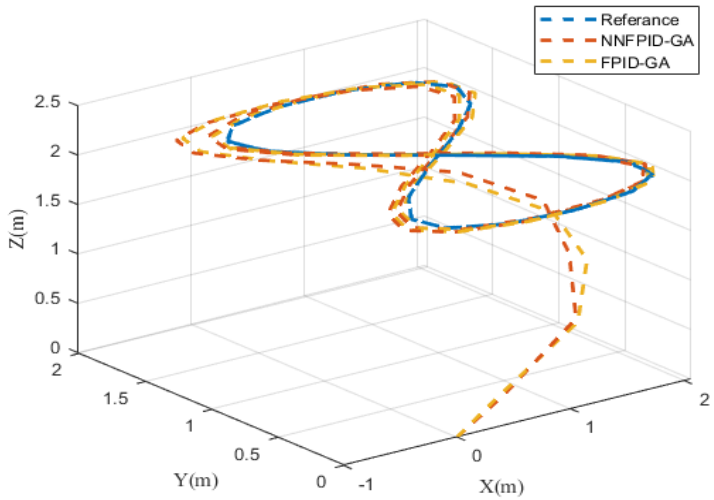


Figure 6.16: 3D Plot for Infinity Trajectory Tracking

6.7 Spiral Trajectory Tracking

In the previous section of Chapter Four, the controllers were designed based on the desired spiral trajectory data. Therefore, in this section, the performance of the controllers for this trajectory was analyzed, and the results were presented in the form of graph plots shown in Figure (6.17),(6.18)and(6.19).

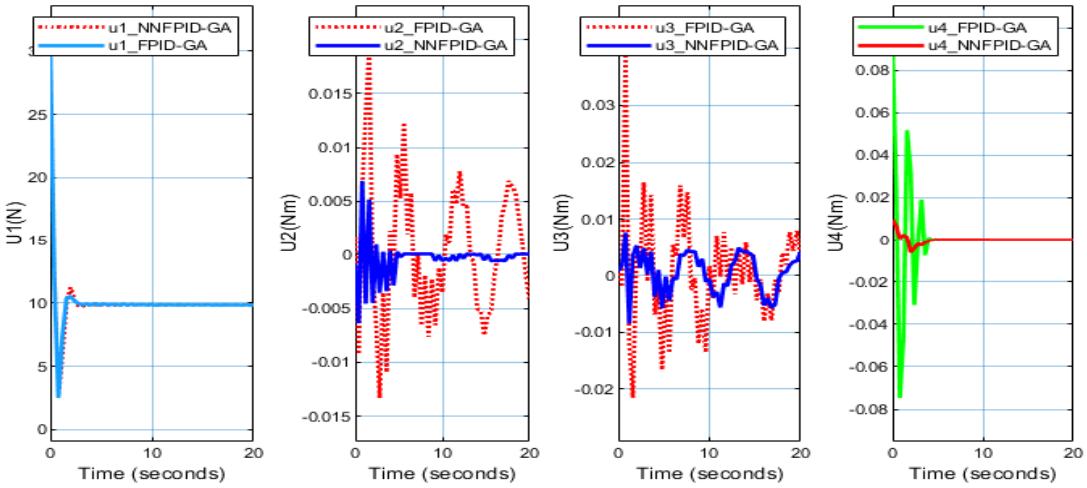


Figure 6.17: Control efforts for Spiral Trajectory Tracking

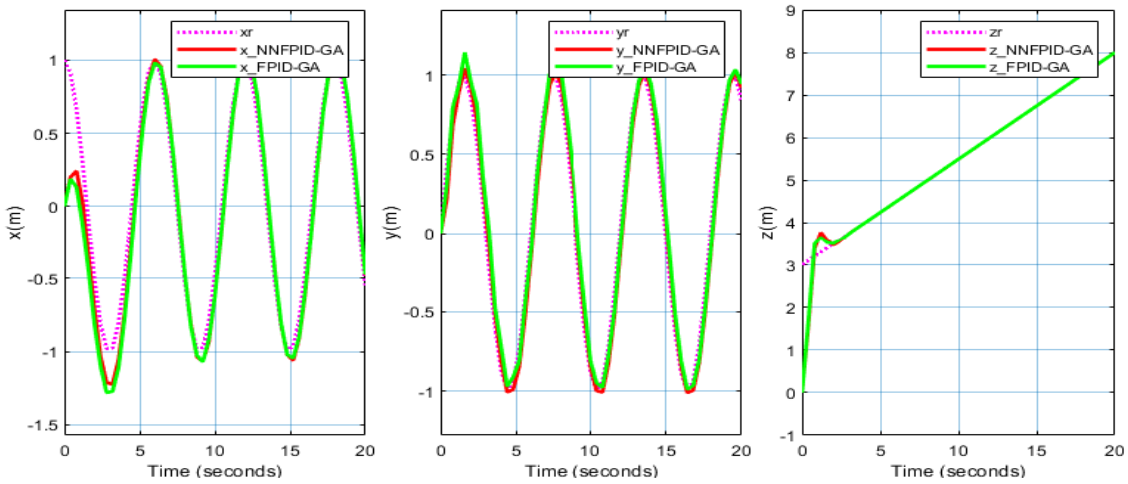


Figure 6.18: Position Tracking Response for Spiral Trajectory Tracking

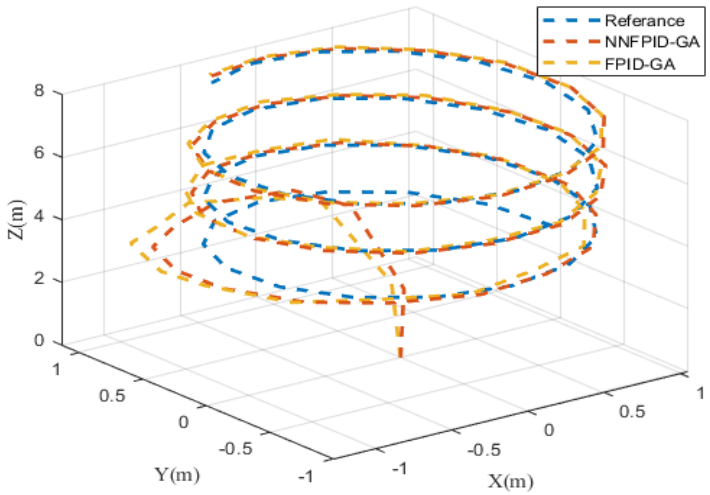


Figure 6.19: 3D Plot for Spiral Trajectory Tracking

6.8 Flight test using UAV Toolbox

In this section of the thesis work, a flight test was conducted using the UAV Toolbox to evaluate the performance of the proposed GA Optimized Fuzzy-PID Based Neural Network controller in a 3D unreal simulation (unreal flight). The GA Optimized Fuzzy-PID Based Neural Network controller was previously designed and validated for trajectory tracking in the preceding section.

The UAV Toolbox, a comprehensive software package for simulation and flight control, played a critical role in enabling real-time assessment of the proposed controller's performance during the flight test in the unreal flight environment.

During the flight test, the UAV was maneuvered along a predefined trajectory similar to the ones used in the trajectory tracking evaluation. The objective was to assess the controller's ability to accurately track the desired trajectory and maintain stable flight throughout.

This research aimed to evaluate the performance of the developed GA Optimized Fuzzy-PID Based Neural Network controller through a flight test conducted using the UAV Toolbox. The flight test proved to be a valuable tool in assessing the performance of the proposed controller for UAVs. The results demonstrated the controller's capability in achieving stable control under different flight conditions. Additionally, the study emphasized the importance of flight testing in validating advanced control algorithms and laid the groundwork for future research to enhance the performance of neural network-based UAV controllers.

Overall, the flight test using the UAV Toolbox was a crucial step in verifying the performance of the developed GA Optimized Fuzzy-PID Based Neural Network controller and ensuring its readiness for real-world applications.

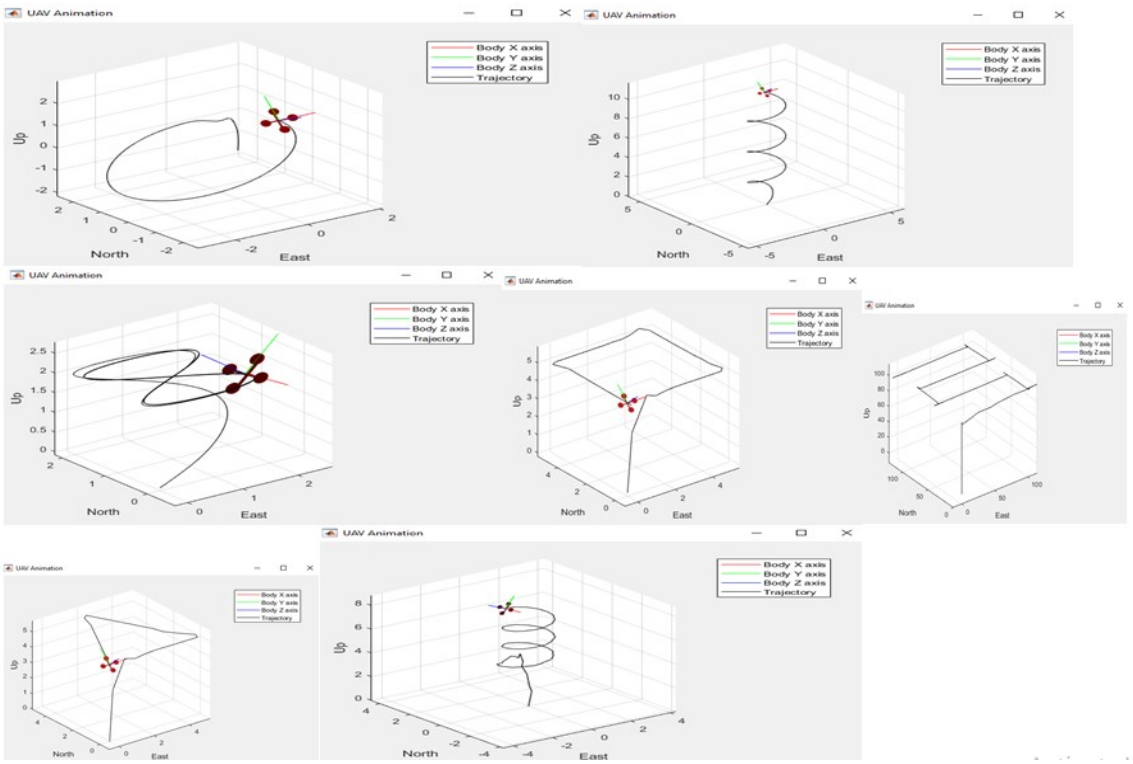


Figure 6.20: Different types of Flight test under GA Optimized Fuzzy-PID Based Neural Network Controller using UAV Toolbox

6.9 Step Response

The previous sections covered a comprehensive study of various types of trajectory tracking control. The performance of the proposed controller and the GA Optimized Fuzzy-PID controller was analyzed using graph plots in Simulink on MATLAB®. This section will focus on discussing the fixed point tracking (step response) of the two controllers. The plots representing this discussion are displayed in Figure (6.21),(**??**),(6.22) and(**??**).

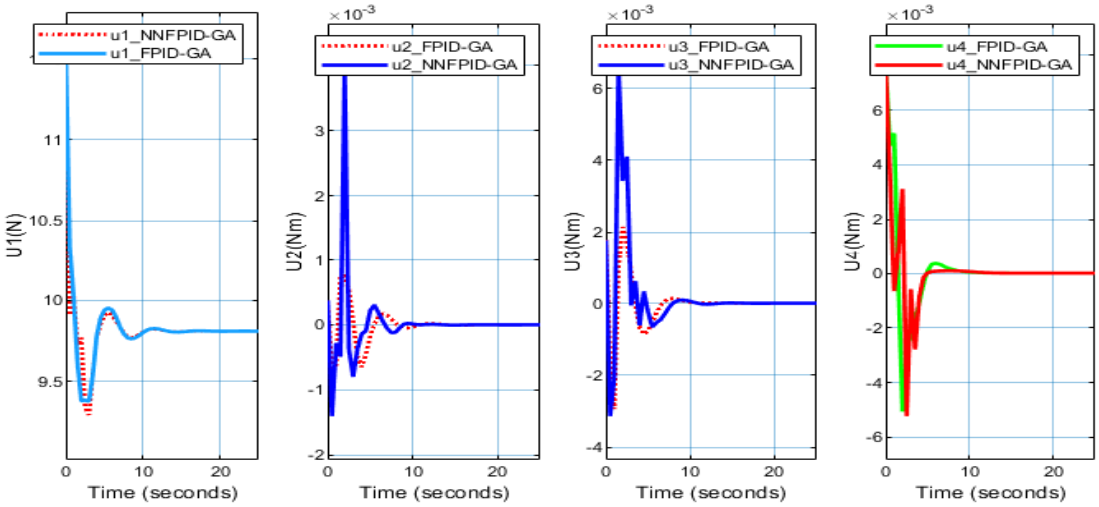


Figure 6.21: Control efforts for Step Response

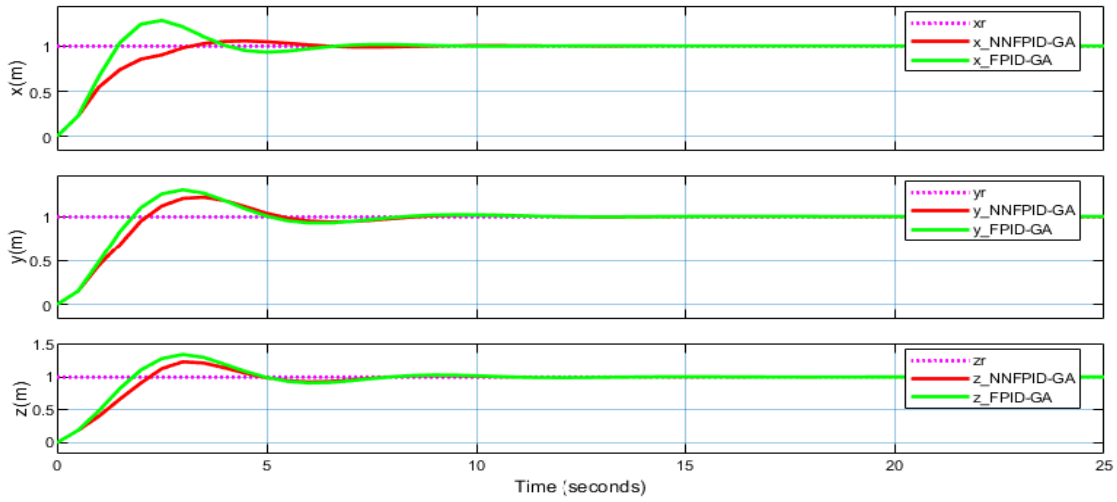


Figure 6.22: Position Response for Step Response

6.10 Parameter Variation and Disturbance Rejection Capability

This section focuses on studying the Disturbance Rejection Capability and handling of Parameter Variation in the proposed and GA Optimized Fuzzy-PID controllers. To simulate real-world conditions, an input disturbance in the form of a random force, representing wind force, was added to the controller inputs at x, y, and z. Additionally, for parameter variation, 25% of the total quad-rotor mass was considered.

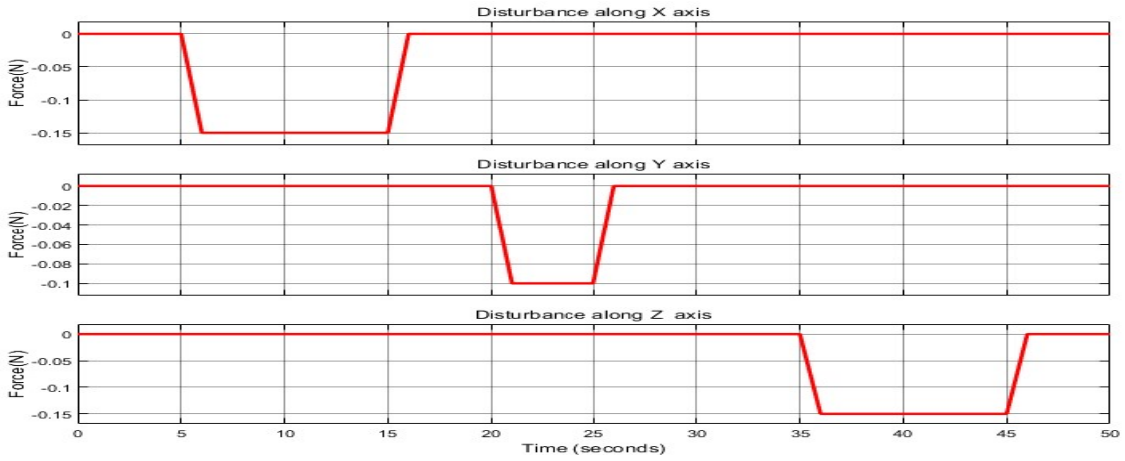


Figure 6.23: The Input Disturbance added along Positions for Rectangular Trajectory Tracking of Quad-Rotor

6.11 Rectangular Trajectory Tracking with Disturbance

This section focuses on the study of Rectangular Trajectory Tracking with an added input disturbance force. The effect of the disturbance force was analyzed using graph plots, with Figure (6.23) showing the added input force. The performance of the GA Optimized Fuzzy-PID and proposed controller (GA Optimized Fuzzy-PID Based Neural Network Controller)

will be further investigated in the later section, specifically in the Comparison of controllers based on Performance metrics.

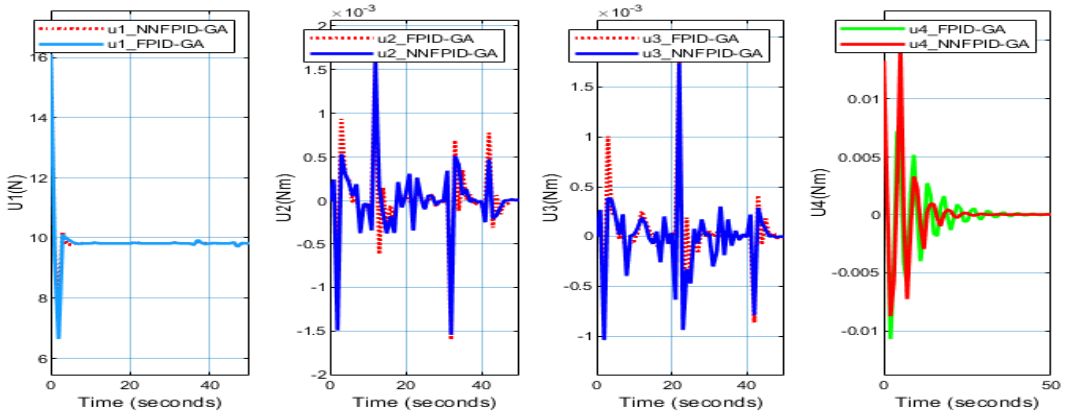


Figure 6.24: Control efforts for Rectangular Trajectory with added Input Disturbance

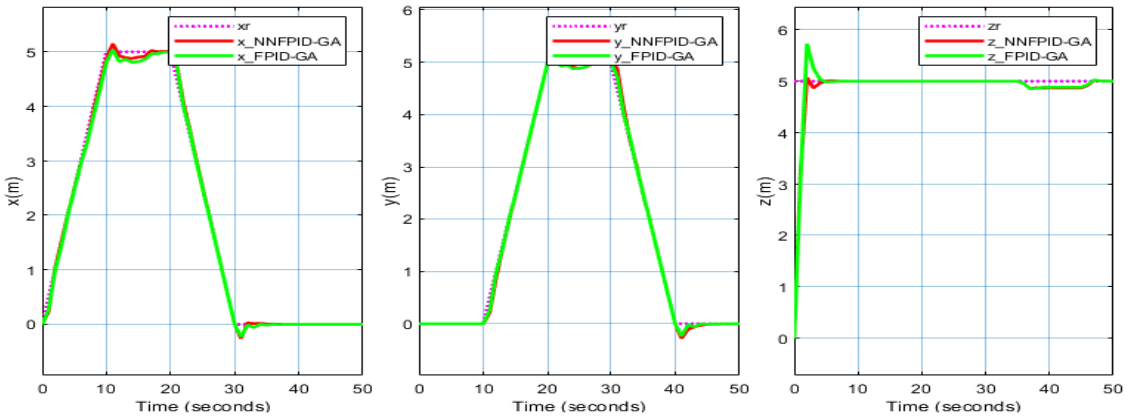


Figure 6.25: Position Tracking Response for Rectangular Trajectory with added Input Disturbance

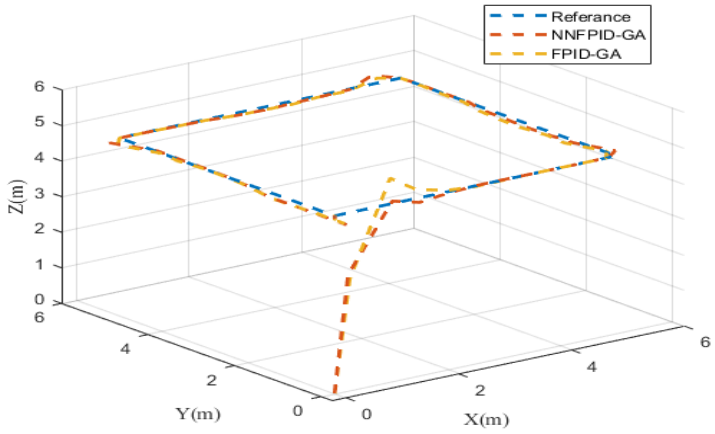


Figure 6.26: 3D Plot for Rectangular Trajectory with added Input Disturbance

6.12 Rectangular Trajectory Tracking with Parameter Variation

This thesis on Parameter Variation examined the effects of increasing the mass of the Quadrotor by 25%, resulting in a total takeoff mass of 1.25kg. The Graph plot for Parameter Variation can be seen below, and the subsequent sections will explore the comparison between the GA Optimized Fuzzy-PID and the proposed controller (GA Optimized Fuzzy-PID Based Neural Network) in terms of their performance metrics.

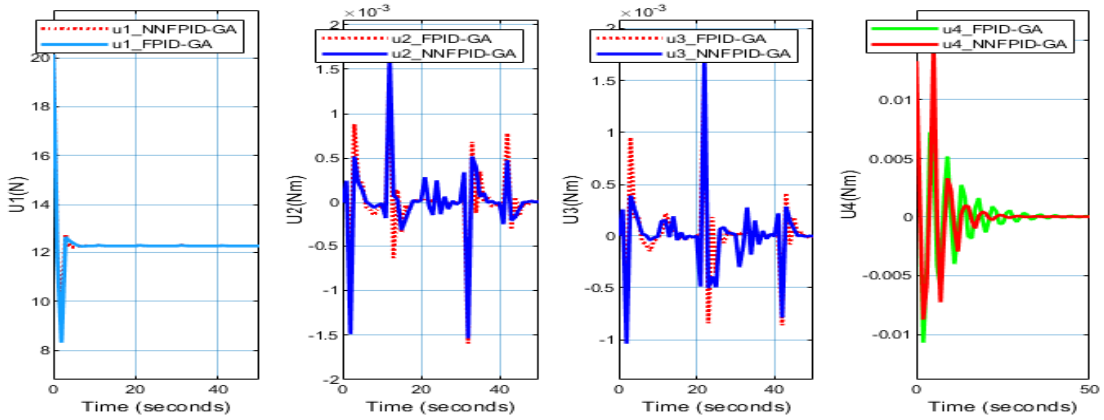


Figure 6.27: Control efforts for Rectangular Trajectory with Parameter Variation

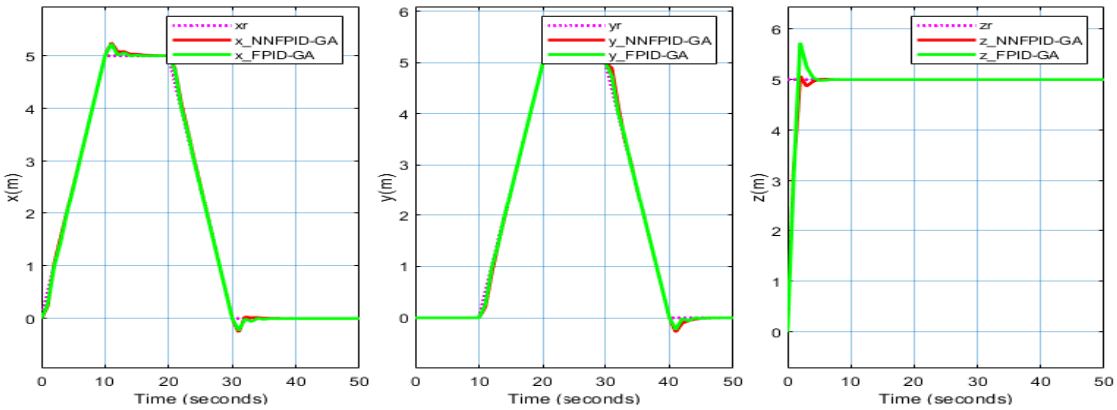


Figure 6.28: Position Tracking Response for Rectangular Trajectory with Parameter Variation

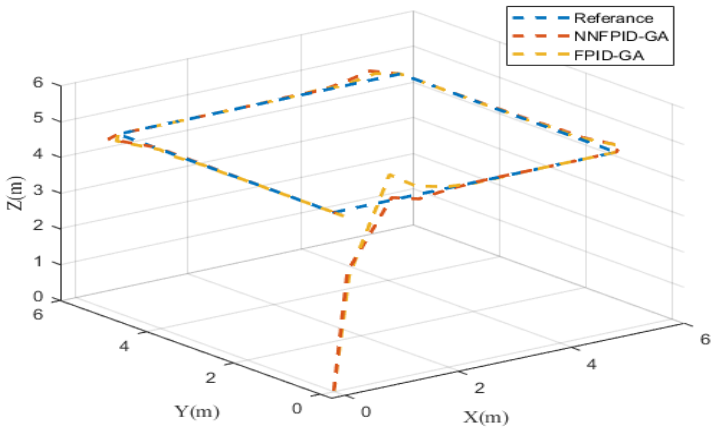


Figure 6.29: 3D Plot for Rectangular Trajectory with Parameter Variation

6.13 Rectangular Trajectory Tracking with Disturbance and Parameter Variation

In this portion, the external force from random disturbances shown in Figure (6.23) was incorporated alongside the control measures for the position and altitude dynamics. The Quad-rotor's mass was adjusted by 25% for parameter variation, resulting in a total mass of m=1.25kg. The outcome of this can be seen in Figures(6.30),(6.31) and(6.33).

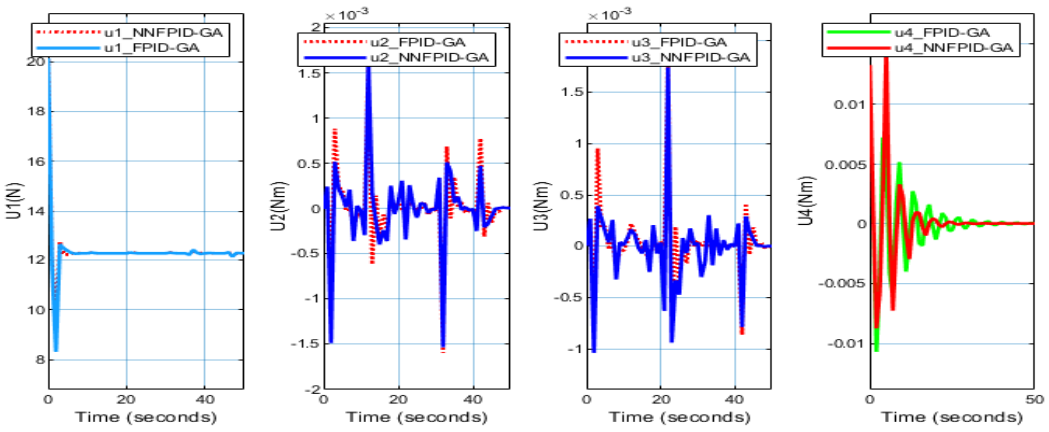


Figure 6.30: Control efforts for Rectangular Trajectory with added Input Disturbance and Parameter Variation

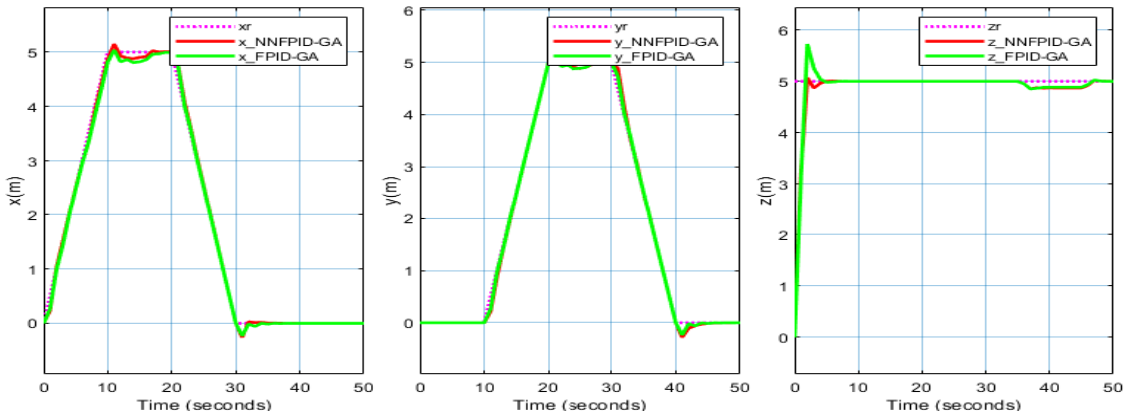


Figure 6.31: Position Tracking Response for Rectangular Trajectory with added Input Disturbance and Parameter Variation

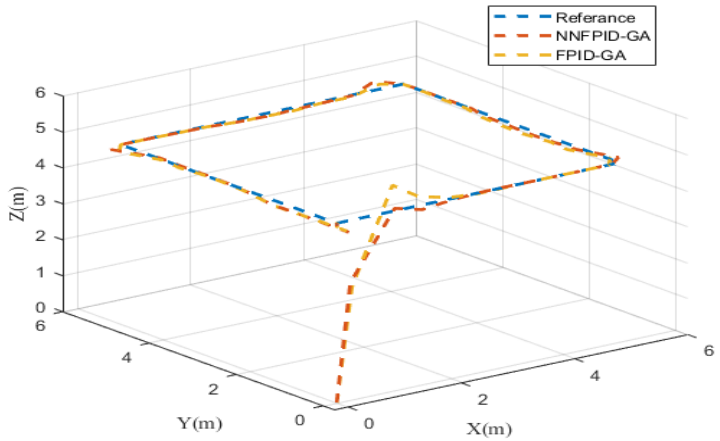


Figure 6.32: 3D Plot for Rectangular Trajectory with added Input Disturbance and Parameter Variation

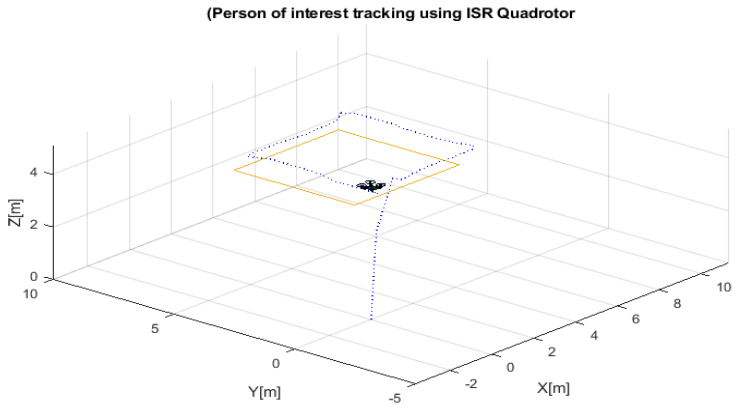


Figure 6.33: (Person of interest tracking where yellow colour indicates person of interest and blue indicates Quadrotor Trajectory)3D Plot for Rectangular Trajectory with added Input Disturbance and Parameter Variation

6.14 Step Response with Disturbance and Parameter Variation

In order to analyze the step response, a random wind force is taken into account as an input disturbance. For disturbance analysis, random time varying forces were applied to the position and altitude control inputs. Additionally, in order to observe the effect of parameter variation, the mass of the Quad-rotor was increased by 25%, as was done for trajectory tracking in the previous section.

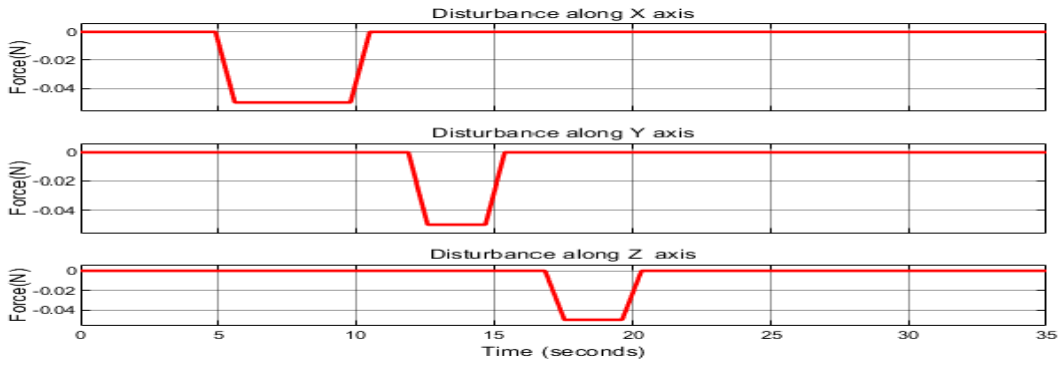


Figure 6.34: The Input Disturbance added along Positions for Step Response

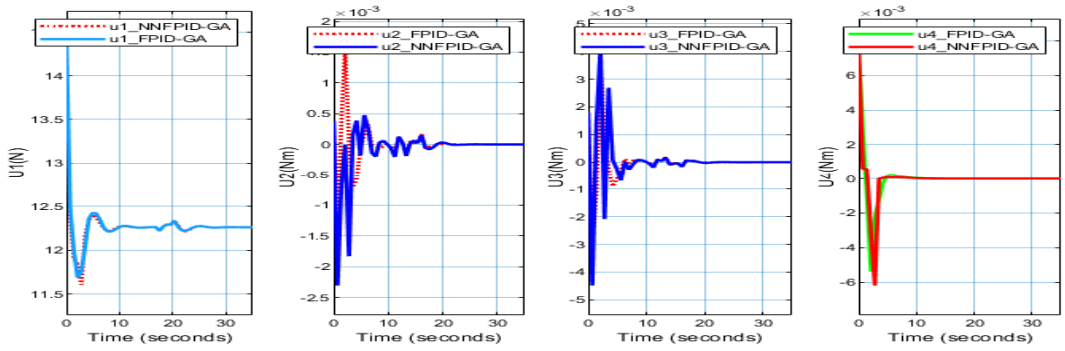


Figure 6.35: Control efforts for Step Response with added Input Disturbance and Parameter Variation

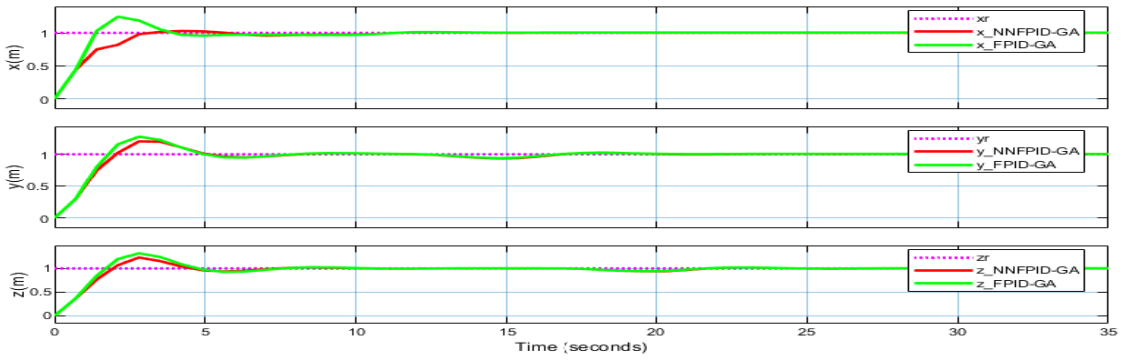


Figure 6.36: Position Tracking Response for Step Response with added Input Disturbance and Parameter Variation

6.15 Comparison of Controllers Based on Performance Indices(Measurement)

In this section, a comparison was conducted between the GA Optimized Fuzzy-PID based NN & the GA Optimized Fuzzy-PID controller, focusing on performance indices. Performance indices refer to the errors commonly used in control engineering to analyze controller performance. In this thesis, the scaling factors were tuned using the Genetic Algorithm, with the cost function being the ITAE (Integral Time Absolute Error). The response to this tuning is illustrated in the section below.

6.16 Comparison of Controllers Based on Performance Indices Using Trajectory Tracking Performance

The performance of controllers in Rectangular Trajectory was evaluated in this section by comparing their trajectory tracking abilities. Performance was assessed through performance indices, as well as testing the effects of input disturbance and parameter variations. The proposed and GA Optimized Fuzzy-PID controllers were both subjected to these performance evaluations in the following subsections.

6.16.1 Comparison of Controllers Based on Performance Indices Using Rectangular Trajectory Tracking Performance

In this subsection, the performance of rectangular trajectory controllers was examined in the absence of disturbances, as well as in the presence of input disturbances and variations in parameters. This was outlined in the performance measurement results shown in the simulation plot above, which will be presented in a tabular format in the following subsections.

Controllers Performance Comparison for Rectangular Trajectory Tracking without applying Disturbance and Parameter Variation

This specific subsection focused on comparing the performance of controllers through simulation. The simulation was conducted without the influence of disturbance force or changes in parameters, resulting in the performance analysis of Rectangular Trajectory.The resulting performance measurements can be found in the table below.

Table 6.1: Controllers Performance Indices for Rectangular Trajectory

	GA Optimized Fuzzy-PID (ITAE(%))	NNFPID-GA (ITAE(%))
phi	30.67	29.66
theta	21.98	21.57
psi	41.94	31.63
x	40.42	38.17
y	71.54	67.01
z	47.42	35.39

Based on the data presented in table(6.1), it can be observed that the rectangular trajectory tracking results without any external input disturbance and parameter variation

show that the GA Optimized Fuzzy-PID controller had an ITAE of 40.42% for the x position, while the GA Optimized Fuzzy-PID Based Neural Network control had an ITAE of 38.17%. The performance indices for the y position were 71.54% and 67.01% for GA Optimized Fuzzy-PID and GA Optimized Fuzzy-PID Based Neural Network, respectively. The altitude (z) position also showed a better performance for GA Optimized Fuzzy-PID Based Neural Network with an ITAE of 35.39%, compared to GA Optimized Fuzzy-PID's ITAE of 47.42%. The roll and pitch angle responses for GA Optimized Fuzzy-PID were 21.98% and 30.67%, while NN had 21.57% and 29.66%, respectively. For the heading angle, GA Optimized Fuzzy-PID had a performance index of 41.94%, while GA Optimized Fuzzy-PID Based Neural Network had 31.63%. Overall, the GA Optimized Fuzzy-PID Based Neural Network controller showed improved tracking performance, with increases of 2.25%, 4.53%, and 12.03% for x, y, and z positions, as well as 1.01% and 10.31% for roll and heading angles, respectively. This suggests that the proposed controller outperforms the GA Optimized Fuzzy-PID controller in terms of tracking performance, as indicated by the performance measurements.

Controllers Performance Comparison for Rectangular Trajectory with input added Disturbance

Table 6.2: Controllers Performance Indices for Rectangular Trajectory Tracking with Input Disturbance

	GA optimized Fuzzy-PID(ITAE(%))	NNFPID-GA (ITAE(%))
phi	31.85	30.61
theta	23.92	23.78
psi	41.94	31.63
x	47.85	45.63
y	63.65	61.53
z	58.87	55.49

The results of rectangular trajectory tracking performance with the addition of external input disturbance are shown in table (6.2). The position x response for the GA Optimized Fuzzy-PID controller had an ITAE measurement of 47.85%, while the GA Optimized Fuzzy-PID Based Neural Network control response was slightly better at 45.63%. For position y, the GA Optimized Fuzzy-PID and NNFPID-GA performance indices were 63.65% and 61.53% respectively. In terms of altitude z, the GA Optimized Fuzzy-PID controller had an ITAE measurement of 58.87%, while the NNFPID-GA controller had a slightly lower measurement of 55.49%. The pitch and roll angle responses were 23.92% and 31.85% for the GA Optimized Fuzzy-PID controller, and 23.78% and 30.61% for the NNFPID-GA controller. The heading angle responses were 41.94% and 31.63% for the GA Optimized Fuzzy-PID and NNFPID-GA controllers respectively. Based on these performance measures, it can be seen that the NNFPID-GA controller improves the tracking performance by 2.22%, 2.12%, and 3.38% for the x, y, and z positions respectively. Similarly, the phi& theta angles improve by 1.24% and 0.14% for the NNFPID-GA controller, while the heading angle improves significantly by 10.31%. Thus, we deduce that the NNFPID-GA controller has better tracking performance compared to the GA Optimized Fuzzy-PID controller based on these performance measures.

Controllers Performance Comparison for Rectangular Trajectory with Parameter Variation

Table 6.3: Controllers Performance Indices for Rectangular Trajectory Tracking with Parameter Variation

	GA Optimized Fuzzy-PID (ITAE(%))	NNFPID-GA (ITAE(%))
phi	30.39	29.39
theta	21.6	21.23
psi	41.94	31.63
x	38.42	33.17
y	57.54	53.01
z	47.42	35.39

Based on the results presented in Table (6.3), the Rectangular trajectory tracking perform ance was evaluated using the GA Optimized Fuzzy-PID controller and the GA Optimized Fuzzy-PID Based Neural Network control. For the x position, the GA Optimized Fuzzy-PID controller showed an ITAE response of 38.42%, while the NNFPID-GA control had a response of 33.17%. Similarly, for the y position, the GA Optimized Fuzzy-PID and NNFPID-GA controllers had performance indices of 57.54% and 53.01%, respectively. The altitude z ITAE response for the GA Optimized Fuzzy-PID controller was 47.42%, compared to 35.39% for the NNFPID-GA controller. The pitch and roll angles also showed slightly higher responses for the GA Optimized Fuzzy-PID controller, with values of 21.6% and 30.39%, compared to 21.23% and 29.39% for the NNFPID-GA controller. The heading angle responses were 41.94% and 31.63% for the GA Optimized Fuzzy-PID and NNFPID-GA controllers, respectively. Overall, the NNFPID-GA controller showed improved performance in all areas, with increases of 5.25%, 4.53%, and 12.03% for the x, y, and z positions, respectively. The roll angle, pitch, and heading angle also saw improvements of 1%, 0.37%, and 10.31%, respectively. Based on these findings, it can be concluded that the NNFPID-GA is more advanced than the GA Optimized Fuzzy-PID controller With regard to tracking performance.

Controllers Performance Comparison for Rectangular Trajectory with input added Disturbance and Parameter Variation

In this particular subsection, the performance of controllers was compared by analyzing the simulation results of the Rectangular Trajectory. The analysis involved applying input disturbance force and parameter variation. The performance measurements obtained from this comparison are provided in the table below.

According to the data in table(6.4), Rectangular trajectory tracking performances were measured with the application of external input disturbance and parameter variation. The Position x response of the GA Optimized Fuzzy-PID controller showed an ITAE value of 47.85%, while the GA Optimized Fuzzy-PID Based Neural Network control response had a slightly lower value of 45.63%. Similar results were observed for position y, with the GA Optimized Fuzzy-PID and NNFPID-GA performance indices being 75.53% and 63.53% respectively. In terms of Altitude z, the GA Optimized Fuzzy-PID controller had an ITAE measurement of 58.49%, while the NNFPID-GA controller showed a lower value of 55.87%.

Table 6.4: Controllers Performance Indices for Rectangular Trajectory Tracking with Input Disturbance and Parameter Variation

	GA Optimized Fuzzy-PID (ITAE(%))	NNFPID-GA (ITAE(%))
phi	31.59	30.58
theta	23.56	23.36
psi	41.94	31.63
x	47.85	45.63
y	75.53	63.53
z	58.49	55.87

The response of the roll and pitch angles for the GA Optimized Fuzzy-PID were 23.56% and 31.59% respectively, while for NNFPID-GA they were 23.36% and 30.58%. The Heading angle controller responses showed values of 41.94% and 31.63% for GA Optimized Fuzzy-PID and NNFPID-GA respectively. Based on these performance measures, it can be seen that the NNFPID-GA controller showed improvements of 2.22%, 12%, and 2.62% for x, y, and z positions respectively, and 1.01%, 0.2%, and 10.31% for roll, pitch, and heading angles. As a result, the proposed controller, the NNFPID-GA controller, showed better tracking performance compared to the GA Optimized Fuzzy-PID controller, even when faced with external input disturbance and parameter variation. This indicates the robustness of both controllers, with the NNFPID-GA controller having the better tracking performance based on the performance measures obtained in this thesis.

Chapter 7

Conclusion and Recommendation for Future Work

7.1 Conclusion

In conclusion, this thesis work focused on the control of an ISR (Intelligence, Surveillance, and Reconnaissance) Quad-rotor UAV using intelligent control technique. The aim was to address the problem of control of a quadrotor's trajectory tracking.

The first step involved deriving a nonlinear mathematical model based on the Newton Euler formalism. To validate the model's accuracy in representing a real Quadrotor, a model verification process was conducted. Once the model was verified, a fuzzy rule-based GA Optimized Fuzzy-PID controller was implemented. The fuzzy logic approach was utilized to adjust the control parameters based on fuzzy rules. The Genetic Algorithm was employed to determine the scaling factors of the Fuzzy-PID controller.

In the next phase, the Proposed control(NNFPID-GA) system was designed using input-output data from the GA Optimized Fuzzy-PID controller. The Neural Network (NN) Toolbox was utilized to design the NNFPID-GA controller. The Levenberg Marquardt backpropag ation algorithm was employed to update the weights of the network. The proposed control algorithm's validity was proven through MATLAB®simulink simulations. The flight test using the UAV Toolbox was conducted to test the stable flight of Quadrotor UAV using developed NNFPID-GA controller. The results demonstrated the controller's capability in achieving stable control under various flight conditions.The goal was to assess the controller's ability to accurately follow the desired trajectory and maintain stable flight throughout. A comparison study based on Performance metrics were conducted to assess the effectiveness of both controllers.

With regard to performance evaluation, even though both the GA Optimized Fuzzy-PID and GA Optimized Fuzzy-PID Based Neural Network(NNFPID-GA) controllers showed improved performance, quicker response and better robustness with regards to settling time, overshoot, disturbance rejection, and handling parameter variation. However, it was observed that the NNFPID-GA controller outperformed the GA Optimized Fuzzy-PID controller.

Lastly, a face recognition system was implemented using Python. The training results exhibited an accuracy of 98.65%, demonstrating the system's ability to detect a wanted person (Person of interest) from other individuals. This thesis work is of significant importance

for law enforcement agencies as it assists commanders in the decision-making process to uphold the rule of law using ISR Quadrotor drone.

Overall, the thesis work successfully achieved its objectives and showcased the effectiveness of intelligent control techniques in addressing trajectory tracking control for ISR quad-rotor while also implementing a face recognition system for law enforcement purposes.

7.2 Recommendation for Future Work

- Developing a control algorithm for obstacle avoidance. (Currently, when flying the closed-loop plant with the controller using MATLAB® SIMULINK UAV toolbox, the drone passes through walls and houses. This indicates that in a real-world scenario, the drone would crash. Therefore, a control algorithm for obstacle detection should be modeled and implemented for future work.)
- Modelling the motor dynamics of the quadrotor to test the performance of the controller, taking into account the motor's behavior.
- Implementing the system using hardware to validate the performance of the developed control algorithm in a real-world environment.

Bibliography

[1] Chee Nam Chua. *Integration of multiple UAVs for collaborative ISR missions in an urban environment*. PhD thesis, Monterey, California. Naval Postgraduate School, 2012.

[2] Koç Mehmet Tuğrul. Drone technologies and applications. 2023.

[3] Colin Wall and John Christianson. Europe's missing piece. 2023.

[4] Wendy Laverick. *Global injustice and crime control*. Routledge, 2016.

[5] Kelley M Sayler. Emerging military technologies: Background and issues for congress. *Washington DC: Congressional Research Service*, 2020.

[6] Vera Lúcia Raposo. The use of facial recognition technology by law enforcement in europe: a non-orwellian draft proposal. *European Journal on Criminal Policy and Research*, pages 1–19, 2022.

[7] Stuart Johann Maxwell Duncan. Visual control of multi-rotor uavs. 2014.

[8] Alireza Manzoori and Gholamreza Vossoughi. Control of quadrotors for tracking and landing on a mobile platform. In *2018 6th RSI International Conference on Robotics and Mechatronics (IcRoM)*, pages 46–52. IEEE, 2018.

[9] Jian Dong and Bin He. Novel fuzzy pid-type iterative learning control for quadrotor uav. *Sensors*, 19(1):24, 2018.

[10] Lele Xi, Xinyi Wang, Lei Jiao, Shupeng Lai, Zhihong Peng, and Ben M Chen. Gto-mpc-based target chasing using a quadrotor in cluttered environments. *IEEE Transactions on Industrial Electronics*, 69(6):6026–6035, 2021.

[11] Sudhir Nadda and A Swarup. On adaptive sliding mode control for improved quadrotor tracking. *Journal of Vibration and Control*, 24(14):3219–3230, 2018.

[12] Luís Martins, Carlos Cardeira, and Paulo Oliveira. Linear quadratic regulator for trajectory tracking of a quadrotor. *IFAC-PapersOnLine*, 52(12):176–181, 2019.

[13] Roohul Amin, Li Aijun, and Shahaboddin Shamshirband. A review of quadrotor uav: control methodologies and performance evaluation. *International Journal of Automation and Control*, 10(2):87–103, 2016.

[14] Wenxing Zhu and Lihui Wang. Adaptive neural network actuator failure compensation scheme for quadrotor trajectory tracking control. In *2022 2nd International Conference on Computers and Automation (CompAuto)*, pages 6–11. IEEE, 2022.

[15] Ahmed Eltayeb, Mohd Fuaad Rahmat, MA Mohammed Eltoum, MH Sanhoury Ibrahim, and Mohd Ariffanan Mohd Basri. Trajectory tracking for the quadcopter uav utilizing fuzzy pid control approach. In *2020 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE)*, pages 1–6. IEEE, 2021.

[16] Khadija El Hamidi, Mostafa Mjahed, Abdeljalil El Kari, and Hassan Ayad. Neural network and fuzzy-logic-based self-tuning pid control for quadcopter path tracking. *Stud. Inform. Control*, 28(4):401–412, 2019.

[17] Mebaye Belete Mamo. Control of quadrotor by designing third-order supertwisting smc. *GSJ*, 8(9), 2020.

[18] Atheer L Salih, M Moghavvemi, Haider AF Mohamed, and Khalaf Sallom Gaeid. Modelling and pid controller design for a quadrotor unmanned air vehicle. In *2010 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)*, volume 1, pages 1–5. IEEE, 2010.

[19] Kaiwen Pan, Yang Chen, Zhengxi Wang, Huaiyu Wu, and Lei Cheng. Quadrotor control based on self-tuning lqr. In *2018 37th Chinese control conference (CCC)*, pages 9974–9979. IEEE, 2018.

[20] Hyeonbeom Lee, Suseong Kim, Tyler Ryan, and H Jin Kim. Backstepping control on se (3) of a micro quadrotor for stable trajectory tracking. In *2013 IEEE international conference on systems, man, and cybernetics*, pages 4522–4527. IEEE, 2013.

[21] Osman Çakira and Tolga Yüksel. Neural network control for quadrotors. *American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS)*, 31(1):191–200, 2017.

[22] Pal Johan From, J Tommy Gravdahl, K Ytterstad Pettersen, et al. *Vehicle-manipulator systems*. Springer, 2016.

[23] Jinhyun Kim, Min-Sung Kang, and Sangdeok Park. Accurate modeling and robust hovering control for a quad-rotor vtol aircraft. In *Selected papers from the 2nd International Symposium on UAVs, Reno, Nevada, USA June 8–10, 2009*, pages 9–26. Springer, 2010.

[24] Luis Rodolfo García Carrillo, Alejandro Enrique Dzul López, Rogelio Lozano, Claude Pégard, Luis Rodolfo García Carrillo, Alejandro Enrique Dzul López, Rogelio Lozano, and Claude Pégard. Modeling the quad-rotor mini-rotorcraft. *Quad Rotorcraft Control: Vision-Based Hovering and Navigation*, pages 23–34, 2013.

[25] Qiong Hu, Qing Fei, Qinghe Wu, and Qingbo Geng. Research and application of nonlinear control techniques for quad rotor uav. In *Proceedings of the 31st Chinese Control Conference*, pages 706–710. IEEE, 2012.

[26] Bora Ly and Romny Ly. Cybersecurity in unmanned aerial vehicles (uavs). *Journal of Cyber Security Technology*, 5(2):120–137, 2021.

[27] Eden Getiye. Model predictive control of unmanned aerial vehicle for locust detection and bio-pesticide spraying,msc thesis aait.

[28] Anežka Chovancová, Tomáš Fico, L'uboš Chovanec, and Peter Hubinsk. Mathematical modelling and parameter identification of quadrotor (a survey). *Procedia Engineering*, 96:172–181, 2014.

[29] N Suthanthira Vanitha, L Manivannan, T Meenakshi, and K Radhika. Stability analysis of quadrotor using state space mathematical modeling. *Materials Today: Proceedings*, 33:4040–4043, 2020.

[30] Aws Abdulsalam Najm and Ibraheem Kasim Ibraheem. Nonlinear pid controller design for a 6-dof uav quadrotor system. *arXiv e-prints*, pages arXiv–1806, 2018.

[31] Ehsan Abbasi, Mohammad Mahjoob, and Reza Yazdanpanah. Controlling of quadrotor uav using a fuzzy system for tuning the pid gains in hovering mode. In *10th Int. Conf. Adv. Comput. Entertain. Technol*, pages 1–6, 2013.

[32] Khadija El Hamidi, Mostafa Mjahed, Abdeljalil El Kari, and Hassan Ayad. Neural network and fuzzy-logic-based self-tuning pid control for quadcopter path tracking. *Stud. Inform. Control*, 28(4):401–412, 2019.

[33] Tianpeng Huang, Deqing Huang, and Dan Luo. Attitude tracking for a quadrotor uav based on fuzzy pid controller. In *2018 5th International Conference on Information, Cybernetics, and Computational Social Systems (ICCSS)*, pages 1–6. IEEE, 2018.

[34] Ehsan Abbasi, Mohammad Mahjoob, and Reza Yazdanpanah. Controlling of quadrotor uav using a fuzzy system for tuning the pid gains in hovering mode. In *10th Int. Conf. Adv. Comput. Entertain. Technol*, pages 1–6, 2013.

[35] Deepak Gautam and Cheolkeun Ha. Control of a quadrotor using a smart self-tuning fuzzy pid controller. *International Journal of Advanced Robotic Systems*, 10(11):380, 2013.

[36] M Zareb, R Ayad, and W Nouibat. Fuzzy-pid hybrid control system to navigate an autonomous mini-quadrotor. In *3rd international conference on systems and control*, pages 906–913. IEEE, 2013.

[37] Sreekanth Damodaran. Genetic algorithms (gas); mimicking the natural selection process to arrive at optimum design solution (s).

[38] OS Eluyode and Dipo Theophilus Akomolafe. Comparative study of biological and artificial neural networks. *European Journal of Applied Engineering and Scientific Research*, 2(1):36–46, 2013.

[39] Jyh-Woei Lin. Artificial neural network related to biological neuron network: a review. *Advanced Studies in Medical Sciences*, 5(1):55–62, 2017.

[40] AD Dongare, RR Kharde, Amit D Kachare, et al. Introduction to artificial neural network. *International Journal of Engineering and Innovative Technology (IJEIT)*, 2(1):189–194, 2012.

[41] OS Eluyode and Dipo Theophilus Akomolafe. Comparative study of biological and artificial neural networks. *European Journal of Applied Engineering and Scientific Research*, 2(1):36–46, 2013.

[42] S Sapna, A Tamilarasi, M Pravin Kumar, et al. Backpropagation learning algorithm based on levenberg marquardt algorithm. *Comp Sci Inform Technol (CS and IT)*, 2:393–398, 2012.

[43] Chen Lv, Yang Xing, Junzhi Zhang, Xiaoxiang Na, Yutong Li, Teng Liu, Dongpu Cao, and Fei-Yue Wang. Levenberg–marquardt backpropagation training of multilayer neural networks for state estimation of a safety-critical cyber-physical system. *IEEE Transactions on Industrial Informatics*, 14(8):3436–3446, 2017.

[44] Rafik Zayani, Ridha Bouallegue, and Daniel Roviras. Levenberg-marquardt learning neural network for adaptive predistortion for time-varying hpa with memory in ofdm systems. In *2008 16th European Signal Processing Conference*, pages 1–5. IEEE, 2008.

[45] Hao Yu and Bogdan M Wilamowski. Levenberg-marquardt training. *Industrial electronics handbook*, 5(12):1, 2011.

[46] Kenneth J Hunt, D Sbarbaro, R Żbikowski, and Peter J Gawthrop. Neural networks for control systems—a survey. *Automatica*, 28(6):1083–1112, 1992.

[47] Abdullah Talha Kabakus. An experimental performance comparison of widely used face detection tools. *ADCAIJ: Advances in distributed computing and artificial intelligence journal*, 8(3):5–12, 2019.

[48] Nikhil Rai, Dhirender Gulair, Jawad Shiningwala, and Mayuri H Molawade. Facial emotion recognition and detection in python using deep learning. *networks*, 8(7), 2021.

[49] M Rovai. Real-time face recognition: an end-to-end project, 2021.

Appendix A

Complete MATLAB®SIMULINK Plant with NNFPID-GA& GA Optimized Fuzzy-PID Controller

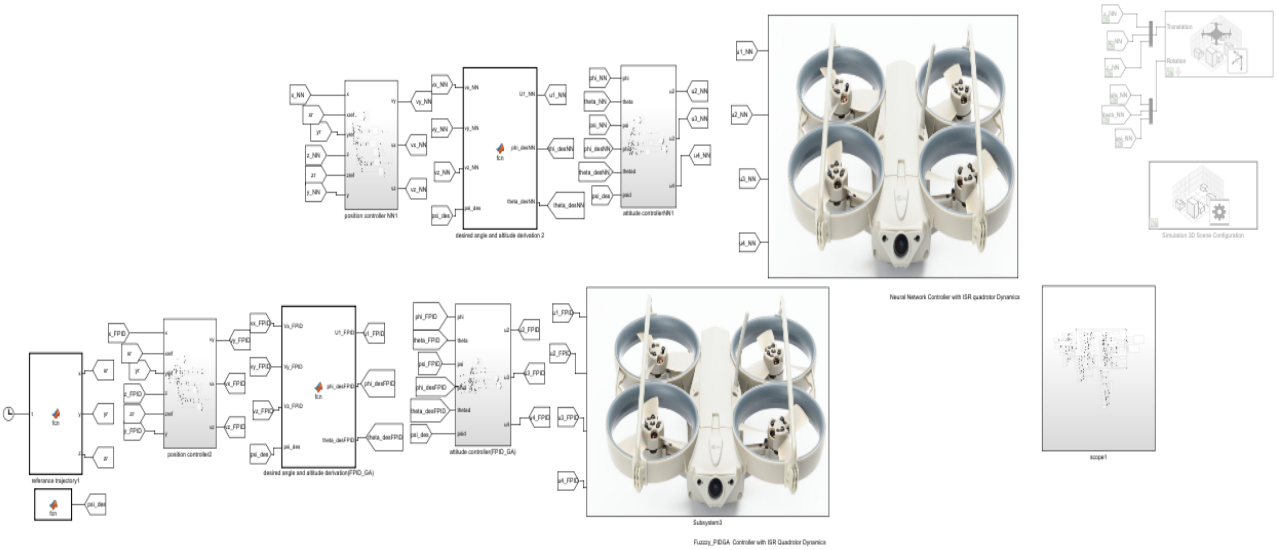


Figure A.1: Complete Plant Model with Controllers

A.1 A Single DOF Control with GA Optimized Fuzzy-PID & NNFPID-GA Controller

In this section the MATLAB®Simulink for the two controllers are described and the way in which the 6DOF Plant Dynamics Control Algorithm was the same hence,the Altitude Dynamics for GA Optimized Fuzzy-PID and Proposed Controllers in Simulinlk as below form and it was identical for other Dynamics(6DOF)the only Difference was the Scaling Parameters which is tuned by using Genetic Algorithm Optimization Methods(GA).

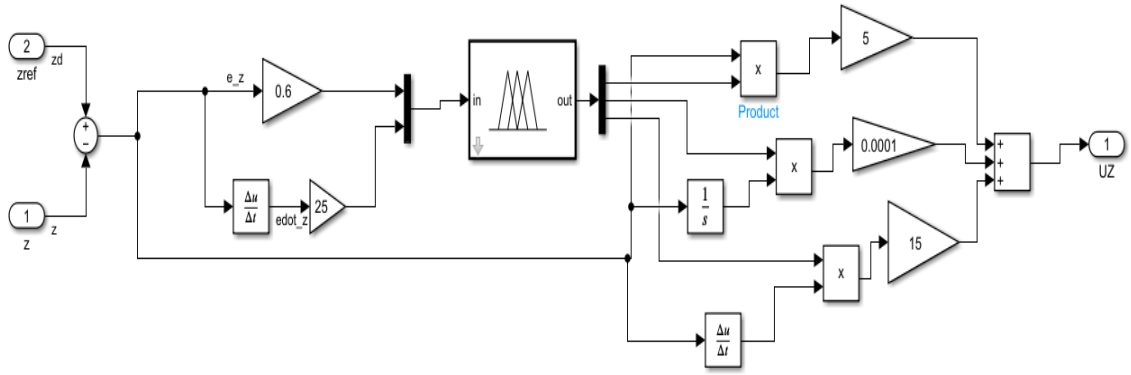


Figure A.2: Altitude Control using GA Optimized Fuzzy-PID Controller

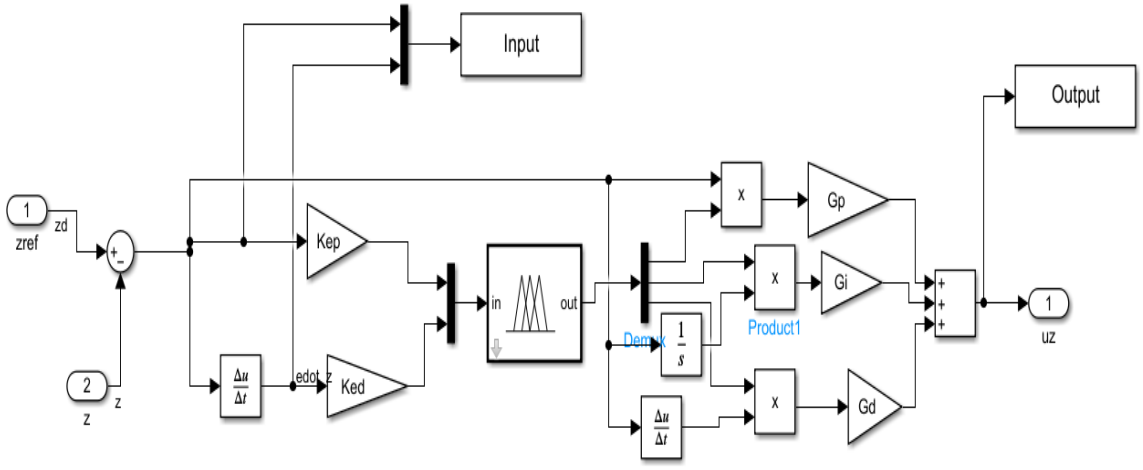


Figure A.3: Input Output Training Data for NNFPID-GA Controller

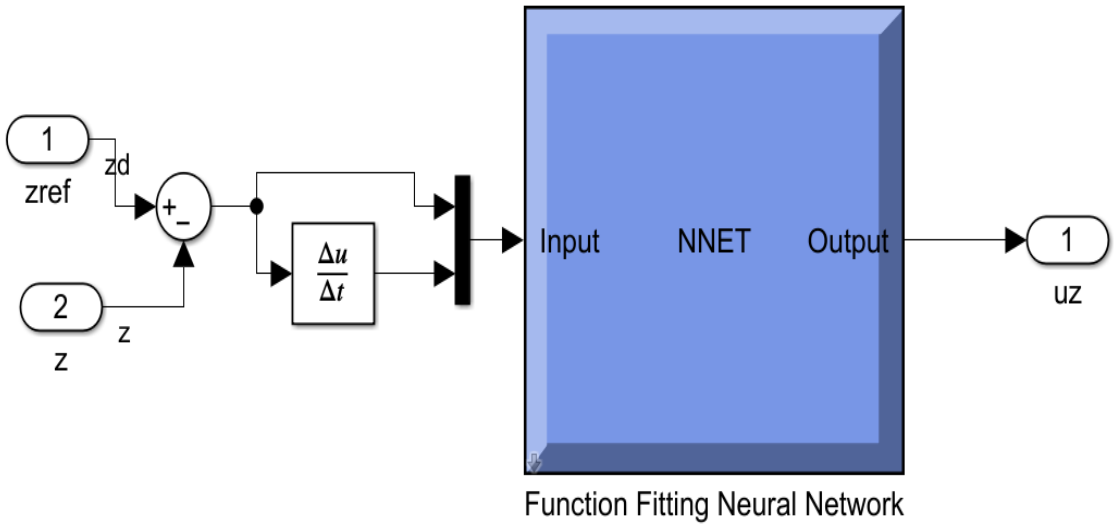


Figure A.4: Altitude Control using NNFPID-GA Controller based on I/O training Data from GA Optimized Fuzzy-PID Controller

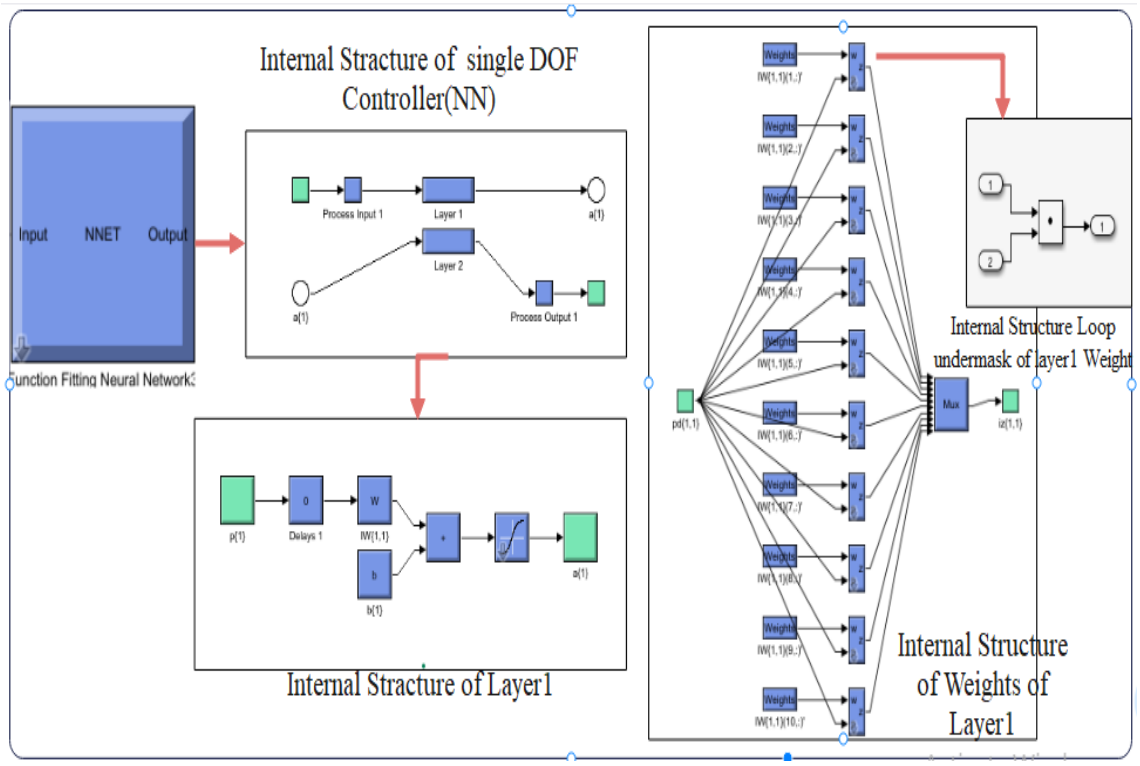


Figure A.5: Altitude Control(NNFPID-GA)Internal Structure

Transfer Matrix

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = T \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{A.1}$$

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = R_{\phi}^{-1} R_{0}^{-1} \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} + R_{\phi}^{-1} \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} \tag{A.2}$$

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ \sin(\phi)\sin(\theta) & \cos(\phi) & \sin(\phi)\cos(\theta) \\ \cos(\phi)\sin(\theta) & -\sin(\phi) & \cos(\phi)\cos(\theta) \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & \sin(\phi) \\ 0 & -\sin(\phi) & \cos(\phi) \end{bmatrix} \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} \tag{A.3}$$

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} -\sin(\theta)\dot{\psi} \\ \sin(\phi)\cos(\theta)\dot{\psi} \\ \cos(\phi)\cos(\theta)\dot{\psi} \end{bmatrix} + \begin{bmatrix} 0 \\ \cos(\phi)\dot{\theta} \\ -\sin(\phi)\dot{\theta} \end{bmatrix} + \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} \tag{A.4}$$

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\sin(\theta) \\ 0 & \cos(\phi) & \sin(\phi)\cos(\theta) \\ 0 & -\sin(\phi) & \cos(\phi)\cos(\theta) \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{A.5}$$

$$T^{-1} = \begin{bmatrix} 1 & 0 & -\sin(\theta) \\ 0 & \cos(\phi) & \sin(\phi)\cos(\theta) \\ 0 & -\sin(\phi) & \cos(\phi)\cos(\theta) \end{bmatrix} \tag{A.6}$$

A.2 Performance Indices

$$\begin{aligned} ISE &= \int_0^{\infty} e^2(t)dt \\ ITSE &= \int_0^{\infty} te^2(t)dt \\ IAE &= \int_0^{\infty} |e(t)|dt \\ ITAE &= \int_0^{\infty} t|e(t)|dt \end{aligned} \tag{A.7}$$

Appendix B

Pseudo Code For Square Wave Trajectory

The square wave trajectory for ISR Quadrotor was generated based on speed information and generating polynomial Trajectory as below form.

$$X(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \tag{B.1}$$

using the abve Polynomial equation we can find the parameters

$$a_0, a_1, a_2 \& a_3$$

from the Plant speed information using initial and final time of the trajectory plan of the ISR mission Quadrotor the result was generated as below.

Algorithm 3 Pseudo code For Square wave trajectory

if $t \leq 10$ then
    $X_{\text{ref}} \leftarrow 0$
    $Y_{\text{ref}} \leftarrow 0$
    $Z_{\text{ref}} \leftarrow 3 * t^2 - (1/5) * t^3$
else if $t > 10$ and $t \leq 20$ then
    $X_{\text{ref}} \leftarrow 500 - 120 * t + 9 * t^2 - 0.2 * t^3$
    $Y_{\text{ref}} \leftarrow 0$
    $Z_{\text{ref}} \leftarrow 100$
else if $t > 20$ and $t \leq 30$ then
    $X_{\text{ref}} \leftarrow 100$
    $Y_{\text{ref}} \leftarrow 0$
    $Z_{\text{ref}} \leftarrow 100$
else if $t > 30$ and $t \leq 40$ then
    $X_{\text{ref}} \leftarrow 100$
    $Y_{\text{ref}} \leftarrow 3240 - 288 * t + (42/5) * t^2 - (4/50) * t^3$;
    $Z_{\text{ref}} \leftarrow 100$
else if $t > 40$ and $t \leq 50$ then
    $X_{\text{ref}} \leftarrow 100$
    $Y_{\text{ref}} \leftarrow 40$
    $Z_{\text{ref}} \leftarrow 100$
else if $t > 50$ and $t \leq 60$ then
    $X_{\text{ref}} \leftarrow$ -32400+1800*t-33*t^2 $+ (1/5) * t^3$
    $Y_{\text{ref}} \leftarrow 40$
    $Z_{\text{ref}} \leftarrow 100$
else if $t > 60$ and $t \leq 70$ then
    $X_{\text{ref}} \leftarrow 0$
    $Y_{\text{ref}} \leftarrow 40$
    $Z_{\text{ref}} \leftarrow 100$
else if $t > 70$ and $t \leq 80$ then
    $X_{\text{ref}} \leftarrow 0$
    $Y_{\text{ref}} \leftarrow 33360 - 1344 * t + 18 * t^2 - (4/50) * t^3$
    $Z_{\text{ref}} \leftarrow 100$
if $t > 80$ and $t \leq 90$ then
    $X_{\text{ref}} \leftarrow 0$
    $Y_{\text{ref}} \leftarrow 80$
    $Z_{\text{ref}} \leftarrow 100$
if $t > 90$ and $t \leq 100$ then
    $X_{\text{ref}} \leftarrow 170100 - 5400 * t + 57 * t^2 - (1/5) * t^3$
    $Y_{\text{ref}} \leftarrow 80$
    $Z_{\text{ref}} \leftarrow 100$
if $t > 100$ and $t \leq 110$ then
    $X_{\text{ref}} \leftarrow 100$
    $Y_{\text{ref}} \leftarrow 80$
    $Z_{\text{ref}} \leftarrow 100$ if $t > 110$ and $t \leq 120$ then
    $X_{\text{ref}} \leftarrow 100$
    $Y_{\text{ref}} \leftarrow 121080 - 3168 * t + 27.6 * t^2 - (4/50) * t^3$
    $Z_{\text{ref}} \leftarrow 100$ if $t > 120$ and $t \leq 130$ then
    $X_{\text{ref}} \leftarrow 100$
    $Y_{\text{ref}} \leftarrow 120$
    $Z_{\text{ref}} \leftarrow 100$ if $t > 130$ and $t \leq 140$ then
    $X_{\text{ref}} \leftarrow -490000 + 10920 * t - 81 * t^2 + (1/5) * t^3$
    $Y_{\text{ref}} \leftarrow 120$
    $Z_{\text{ref}} \leftarrow 100$
else
    $X_{\text{ref}} \leftarrow 0$
    $Y_{\text{ref}} \leftarrow 120$
    $Z_{\text{ref}} \leftarrow 100$
end if

Appendix C

C.1 Genetic Algorithm to tune Scaling Factors

```
%genetic algorithm to tune Scaling  parameters
%initialization
no_var=30;
lb=[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];
ub=[200 200 200 200 200 200 200 200 200 200 200 200 200 200 200
    5 5 5 5 5 5 5 5 5 5 5 5 5 5 5];
%GA options
ga_opt=gaoptimset('Display','off','Generations',100,'
   PopulationSize',100,'plotfcns',@gaplotbestf);
obj_fn=@(k) optimization_PID(k);
%GA command
[k,best]=ga(obj_fn,no_var,[],[],[],[],lb,ub,[],ga_opt);
```

C.2 Fitness Function for GA

```
function cost=optimization_PID(k)
assignin('base','k',k);
sim('nigatu.slx');
cost=ITAE(length(ITAE));
end
```

C.3 Function Main for Face Recognition

```
from recognition import FaceRecognition
# pip install cmake dlib==19.224
if __name__ == '__main__':
    fr = FaceRecognition()
    fr.run_recognition()
```

C.4 Phyton Code for Face Recognition

```
import face_recognition
import os, sys
```

```
import cv2
import numpy as np
import mat
video_capture=cv2.VideoCapture(0)
address = 'http://10.40.0.105:8080/video'
video_capture.open(address)
# Helper
def face_confidence(face_distance, face_match_threshold=0.6):
    range = (1.0 - face_match_threshold)
    linear_val = (1.0 - face_distance) / (range * 2.0)
    if face_distance > face_match_threshold:
        return str(round(linear_val * 100, 2)) + '%'
    else:
        value = (linear_val + ((1.0 - linear_val) * math.pow((
           linear_val - 0.5) * 2, 0.2))) * 100
        return str(round(value, 2)) + '%'
class FaceRecognition:
    face_locations = []
    face_encodings = []
    face_names = []
    known_face_encodings = []
    known_face_names = []
    process_current_frame = True

    def __init__(self):
        self.encode_faces()

    def encode_faces(self):
        for image in os.listdir('faces'):
            face_image = face_recognition.load_image_file(f"
               faces/{image}")
            face_encoding = face_recognition.face_encodings(
               face_image)[0]
self.known_face_encodings.append(face_encoding)
self.known_face_names.append(image)
print(self.known_face_names)
def run_recognition(self):
if not video_capture.isOpened():  sys.exit('Video source not
   found...')
        while True:
            check, frame = video_capture.read()
  # Only process every other frame of video to save time
 if self.process_current_frame:
  # Resize frame of video to 1/4 size for faster face
     recognition processing
small_frame = cv2.resize(frame, (0, 0), fx=0.25, fy=0.25)
 # Convert the image from BGR color (which OpenCV uses) to RGB
    color (which face_recognition uses)
 rgb_small_frame = small_frame[:, :, ::-1]
# Find all the faces and face encodings in the current frame of
```

```
    video
self.face_locations = face_recognition.face_locations(
   rgb_small_frame)
    self.face_encodings = face_recognition.face_encodings(
       rgb_small_frame, self.face_locations)
     self.face_names = []

    # Calculate the shortest distance to face
        face_distances = face_recognition.face_distance(self.
           known_face_encodings, face_encoding)
         best_match_index = np.argmin(face_distances)
        if matches[best_match_index]: name = self.
           known_face_names[best_match_index]
            confidence = face_confidence(face_distances[
               best_match_index])
self.face_names.append(f'{name} ({confidence})')
self.process_current_frame = not self.process_current_frame
     # Display the results
     for (top, right, bottom, left), name in zip(self.
        face_locations, self.face_names):
          # Scale back up face locations since the frame we
             detected in was scaled to 1/4 size
                top *= 4
                right *= 4
                bottom *= 4
                left *= 4
      # Create the frame with the name
     cv2.rectangle(frame, (left, top), (right, bottom), (0, 0,
        255), 2)
      cv2.rectangle(frame, (left, bottom - 35), (right, bottom)
         , (0, 0, 255), (255, 255, 255), 1)
    # Display the resulting image
     cv2.imshow('Face Recognition', frame)
 results = imshow('Face Recognition')
     results = imshow('xy')
    # Hit 'q' on the keyboard to quit!
            if cv2.waitKey(1) == ord('q'):
                break
        # Release handle to the webcam
        video_capture.release()
        cv2.destroyAllWindows()
if __name__ == '__main__':
    fr = FaceRecognition()
    fr.run_recognition()
```