Addis Ababa University

Addis Ababa Institute of Technology

School of Electrical and Computer Engineering

Object Detection and Optimal Fuzzy-PID Controller Design for Tracking

By: Yaregal Limenih Melese

Advisor: Dr. Mengesha Mamo

A Thesis Submitted to Addis Ababa Institute of Technology, School of Graduate Studies, Addis Ababa University

in Partial Fulfillment of the Requirement for the Degree of Master of Science in Control Engineering

Addis Ababa, Ethiopia

October 2021 GC

Addis Ababa University

Addis Ababa Institute of Technology

School of Electrical and Computer Engineering

This is to certify that the thesis prepared by Yaregal Limenih, entitled: *Object Detection and Optimal Fuzzy-PID Controller Design for Tracking*, and submitted in partial fulfilment of the requirements for the degree of Master of Science in Control Engineering complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Approved by Board of Examiners:

Dr. Mengesha Mamo		
Thesis Advisor	Signature	Date
Dr. Lebsework Negash		
Internal Examiner	Signature	Date
Mr. Mesfin Tilahun		
External Examiner	Signature	Date
Dr. Bisrat Derebssa		
School Chair Person	Signature	Date

Declaration

I, the undersigned, confirm that this thesis is my original work. It has not been presented for a degree program in this or other Universities. All sources of materials used have been properly acknowledged and cited.

Yaregal Limenih Melese		
Name	Signature	Date

Place: Addis Ababa Institute of Technology, Addis Ababa University, Addis Ababa

Submission Date: October 11, 2021 GC

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

Dr. Mengesha Mamo		
Advisor's Name	Signature	Date

Acknowledgement

First, I would like to thank the almighty God and his mother St. Marry for everything I have, strengthening me, and showing the path being with me as always throughout my ups and downs. I would like to express my sincere thankfulness to my advisor Dr. Mengesha Mamo (Associate Professor, Power Electronics and Electrical Drives, Addis Ababa University), *a good father*, for his invaluable guidance, support, and motivation during the whole course of the thesis work. Next, I would like to thank Dr. Dereje Shiferaw for his comments and advice during the seminar presentations. His comments are vital for shaping the contents of the thesis. Finally, I want to thank my family, friends, and colleague Yared Tadesse for their invaluable love, patience, encouragement, and comments.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

2-D	Two Dimensional
AC	Alternative Current
AI	Automated Inspections
ANFIS	Adaptive Neuro-Fuzzy Inference System
AVI	Audio Video Interleave
CCD	Charge Coupled Device
CoA	Center of Area
DC	Direct Current
EMF	Electromotive Force
FLC	Fuzzy Logic Control
FOV	Field of View
FPID	Fuzzy Proportional Integral Derivative
GA	Genetic Algorithm
GigE	Gigabit Ethernet
HMI	Human Machine Interface
IEEE	Institute of Electrical and Electronics Engineering
IMAQ	Image Acquisition
LabVIEW	Laboratory Virtual Instrument Engineering Workbench
LED	Light Emitting Diode
MAX	Measurement and Automation Explorer
MF	Membership Function
MIMO	Multiple Input Multiple Output
MIT	Massachusetts Institute of Technology
MV	Machine Vision
myRIO	my Reconfigurable Input Output
NI	National Instruments
OCR	Optical Character Recognition
PID	Proportional Integral and Derivative
PWM	Pulse Width Modulation
RGB	Red Green Blue
RT	Real Time
SI	International System
SISO	Single Input Single Output
T&M	Test & Measurement
USB	Universal Serial Bus
VI	Virtual Instrument
Z-N	Ziegler-Nicholas

List of Symbols

η_g	Motor efficiency
E_b	Back EMF
I_a	Armature current
K_T	Motor torque constant
K_b	Back EMF constant
K_c	Controller gain
K_d	Derivative gain
K_g	Gear box ratio
K_i	Integral gain
K_p	Proportional gain
L_a	Armature winding inductance
R_a	Armature winding resistance
T_M	Motor torque
T_d	Derivative controller gain
T_i	Integral controller gain
T_u	Period of oscillation
T_w	Disturbance torque
V_a	Armature voltage
$r_{E,M}$	Roots electrical/ mechanical
t_r	Rise time
t_s	Settling time
μ_F	Degree of membership function
$\tau_{m,e}$	Time constant mechanical/electrical
D	Viscous-friction coefficient
d	Disturbance
E	Error
F	Fuzzy set
J	Moment of inertia
N	Number of membership functions
R	Rule, Reference
T	Time
U	Controller output, Universe of discourse
X	Element of the universe of discourse
ω	Angular velocity
PB_u	Controller proportional band
θ	Angular position

Abstract

In this thesis, static and dynamic camera-based object tracking systems have been developed. The static camera-based object tracking system was developed using LabVIEW and Math Script node. The dynamic camera-based object tracking system has been proposed using a Fuzzy-PID controller to locate and expeditiously follow the course/ trajectory of the moving object. The pan/tilt mechanism (a servomechanism) has been used to adjust the pan/azimuth and tilt/elevation positioning of the camera setup. PID, Fuzzy and Fuzzy-PID controllers have been designed for comparison purposes. A genetic algorithm was implemented in order to obtain the optimal values of the Fuzzy-PID controller parameters. The corresponding objective function used for optimization was Integral of Time-weighted Absolute Error (ITAE). The simulation results showed that the pan and tilt system time responses recorded for the Fuzzy-PID controller were 9.4 and 7.9 milliseconds rise time, and 16 and 15 milliseconds settling time respectively. In both cases the steady-state error and the maximum overshoot were negligible i.e., steady-state was reached. The overall performance shows that GA tuned Fuzzy-PID controller has given us the best results in contrast to the other control techniques.

Keywords: Object Tracking, LabVIEW, Fuzzy-PID, Pan/Tilt System, Genetic Algorithm.

Chapter One

Introduction

1.1 Background

All animals need to recognize the dynamic nature of their surrounding environment [1]. They do detect and manifold those environmental changes through physiological systems placed in their body. The five sense organs used by animals to understand the scene and take the right actions to reach their goals are sight, hearing, touch, smell, and taste. Animals other than humans (non-rational animals) intentions are just simple and limited to safety, chasing for food and defense. But humans have a high level of understanding and are capable of interpreting the changes happening in the environment, adding knowledge, and think forward to do in advance for future behaviors [1].

Interest object detection and tracking are significant researches in the field of computer vision [2]. Computer vision [3] is the study of letting computers see, recognize and further interact with the world in the way we human beings do. In order to understand and cooperate with the world, computers must be talented to locate objects of interest and continuously follow these objects. This is the task of visual tracking which is also the heart of numerous computer vision applications such as surveillance, robotic, and monitoring applications [1, 2].

The first computer vision-connected research works began in the early 1970s [4]. Researchers tried to impersonate human intelligence into computers. And during then, computer vision was considered as a visual insight component. By feeding a computer some visual inputs, it was expected to define what it sees. The summer vision project proposed by the undergraduate student Gerald J. Sussman of MIT was taken as an example. By linking a camera, a computer was aimed to define what it saw in the surrounding environment [4].

Computer vision systems do not need all sensitive senses rather than sight [1]. They are equipped only with "eyes" and attempt to understand the surrounding environment looking for scene modifications [1]. Scene changes can be occurred due to illumination changes, noise in the acquisition process, and objects movement. The behavior and movement of the object are the most significant features to be traced, due to the fact that illumination changes are global properties and could not give enough information about the dynamic nature of the scene. Noise is an unwanted signal added, by the acquisition device, over the real representation of the scene.

In the extraction of objects' motion in an image, illumination changes and noise should be considered with great care as their change causes it difficult [1].

Object tracking, the main objective of this thesis, is the method of locating a moving object (or several ones) and estimating its course along time in a sequence of images. It comprises two steps: detection of interest objects and traces their trajectories. Furthermore, object tracking could also involve behavior analysis, object and activity classification, specific person identification, counting, flux statistics, and alarming. Tracking [5] is the way of locating a moving object or multiple objects over a period of time with a camera. Object tracking is technically a problem of reckoning object paths in image planes while moving around a scene.

Normally, tracking is seen as the main task involving several subtasks such as image segmentation for object detection, object matching, and position estimation. Lots of tracking algorithms have been established to implement and solve these subtasks. But each of the algorithms has its strengths and weaknesses. To find an optimal tracking system for a certain kind of application, extensive researches have been conducted in this field over the last years. Even though different object tracking approaches have been proposed in many works of literature for the last years, they are not capable to generate accurate results for all kinds of scenes. Good results are dependent on a certain number of assumption verifications [1]. This reason is the motivation to study and implement new tracking approaches introducing new concepts, such as the Fuzzy-PID controller, to improve the tracking process.

LabVIEW, a graphical programming language first brought up for the Apple Macintosh in 1986, can be utilized for many applications like computer vision, data acquisition, industrial automation, instrumental control, and different computations [13]. Image processing using LabVIEW gives better results. It takes place with block representations and becomes simpler than writing complicated programming. Object tracking, pattern matching, edge detection, and histogram, etc. can be easily done using the NI-LabVIEW and NI-Vision Assistant [13, 23].

This thesis work is a great tool to learn about image processing and servo control mechanisms using NI-LabVIEW and GA tuned Fuzzy-PID controller and can be utilized for many applications so that it can greatly reduce very tedious and repetitive human-based operations.

1.2 Problem Statement

Besides its need, tracking an object from a video has made outstanding improvements in many aspects. Despite this, object tracking and its process is a very difficult task and remains a challenge for researchers. The need for efficient and relatively simple object tracking methods also increases drastically.

Many object tracking algorithms have been proposed to detect and track the moving object from the scenes, but those algorithms require more time to process and be effective, and become more complex. Complex algorithms, long design and computation time, increased cost, non-efficient results and unsatisfactory real-time processing requirements are the challenges.

The presented work goal is to study the problem of object tracking in a sequence of images using Fuzzy-PID control concepts. The object must have a distinguishing feature that can be used to characterize it. In this work, only the value of grey level (histogram) is used to distinguish the object from the background. With the introduction of GA tuned Fuzzy-PID controller, optimal tracking of the moving object is expected.

Therefore, LabVIEW-based object tracking is used to eliminate and replace the complicated programming language with a simple drag and drop graphical programming language. The genetic algorithm tuned Fuzzy-PID controller here is used to optimally track the object of interest by controlling the pan/tilt mechanism which is governed by servo systems.

1.3 Objectives of the Thesis

1.3.1 General Objective

The main objective of this thesis is Object Detection and Design of a Genetic Algorithm Tuned Fuzzy-PID Controller for Tracking.

1.3.2 Specific Objectives

The specific objectives of the thesis are:

i. Object detection and tracking in LabVIEW.
ii. To build a modeling simulation for DC servo-motors using NI Control Design and Simulation Module.
iii. To design PID, Fuzzy Logic, and Fuzzy-PID controllers for the pan/tilt mechanism.
iv. To develop a genetic-algorithm based parameter optimization for both the conventional

PID and Fuzzy-PID controllers.

v. Over all model simulation and evaluation.

1.4 Contribution of the Thesis

This thesis contributes the use of genetic algorithm tuned PID and Fuzzy-PID controller methods for moving object course tracking with a dynamic camera configuration. The use of LabVIEW NI-Vision for the static camera-based moving object trajectory tracking is also another contribution of this thesis.

1.5 Organization of the Thesis

This thesis is organized into five chapters. The *first chapter* of the thesis presents the background section of the study, problem statement, objective, and contributions of the study. *Chapter two* presents background and literature review, where a detailed review on object detection and tracking is made. LabVIEW and its different features which are required for the tracking system are also discussed. Furthermore, PID and Fuzzy-Logic controllers are discussed in detail. In the literature review section of this chapter, a comparison of this research and the most related recent research works are discussed. In *chapter three*, the methodology followed in LabVIEW NI-Vision for object detection and tracking, and modeling and parameter selection of the actuator is presented. *Chapter four* presented the design and analysis of PID and Fuzzy-PID controllers and genetic algorithm optimization for object tracking systems. The result and discussion section of the thesis is presented in *chapter five*. Simulation results are presented both in graphical and tabular forms and a brief discussion is made based on the obtained results and the objective of the study. Finally, the last chapter of the thesis, *chapter six*, mentions the conclusion and future works for future study. Cited *references* and supporting *appendices* are noted at the end of this thesis report.

Chapter Two

Background and Literature Review

2.1 Object Tracking and Detection

Being part of computer vision, object tracking is a very difficult and complex research area but has many practical applications. The complexity of the tracking method arises due to many problems like noise, quick lighting condition changes, sudden or complex object motions, change in appearance patterns of the object and the view, non-rigid object appearances, object-to-object, and object-to-view occlusions, camera motion, and real-time processing requirements [1, 4].

Some assumptions are made in almost all object tracking algorithms depending on the type of application needed to make tracking algorithms to minimize their complexity. For instance, the assumption taken looks like object motion is smooth, or impose some constraints on the object moves to make it constant in velocity or acceleration [4]

Human-computer collaboration became more effective due to the emerging technology of object tracking. Computers are allowed to get an improved model/ prototype of the real world. For example, in areas where human communication is difficult to state the environment in a quick and accurate way, autonomous vehicles can be used instead and generate reliable, exact, and effective results [4].

Object tracking is done by using sensor measurement results; for instance, the sensors like camera, radar, sonar, microphone, infrared sensor, ultrasound, or any other sensor types can be used [6]. These sensor measurements can be utilized to gather information data about objects surrounding the environment, to find out the position, course, and features of objects of interest. Based on the data obtained from the sensors, tracking algorithms are employed in order to generate some essential information from the scene.

Object detection involves locating objects in the frame of a video sequence. Every tracking algorithm needs an object-detection system either in each frame or when the object first appears in the video [5].

The three main steps used by the tracking process are detection of object of interest, tracking of those interest objects, and examination of object tracks to understand and interpret their

behavior. Tracking algorithms are capable of identifying objects in video images and follow them in successive sequential video frames. They can identify and locate target objects while ignoring similarly shaped objects. This is in order to track the object's trajectory, measure the speed, and/or investigate the object's interaction with other objects [6, 7].

2.1.1 Application

The early tracking algorithms were mainly used in surveillance tasks, but nowadays researchers made tracking algorithms applicable in a wider range of fields such as [1]: Video surveillance, Human-machine interaction (HMI), Traffic monitoring, Medical support, Live sport video analysis, Laboratory animal tracking, Smart rooms, Robotic vision, and Air space monitoring, etc.

An important tracking problem is the tracking of aircraft with radar, such as for air traffic control [6]. Military surveillance systems use radar tracking in order to identify aircraft's type, speed, location, and the likelihood of intentions whether they are a threat or not. Those radars can have a wide variety of measurement capabilities ranging from simple range measurements to high-resolution imaging. Radar uses reflected radio waves to measure the direction, distance, and radial speed of the detected object. A radar transmitter emits electromagnetic waves, which are reflected by the object and detected by a receiver. The measured data are used to draw out paths, which are often presented together with the object reflections on a display screen [6].

Nowadays, video surveillance is used in all sectors of society such as airports, buildings, banks, department stores, casinos, railway stations, highways, streets, stadiums, crowd gathering places, and all government institutions, as a means to increase public safety and security and deter criminal acts [6].

In order to provide weather forecasts, weather bureaus release weather balloons and track them, which then provide information on high-altitude wind velocities, pressure, humidity, and temperature [6].

In addition, in the examination of cell biology medical researchers carefully organize and capture sequential images at a regular interval where in some cases the images can be collected for several days. The collected images are then analyzed manually in order to find the required parameters such as speed, division, and death of each cell [6]. This process is a very time-consuming task and may lead to making an error, sometimes it became impractical too. And

therefore, in order to escape from such difficulty, this problem can be formulated as the problem of tracking cells over a sequence of images. The events of cell division and death are then observed by looking at the track initiation and termination probabilities [6].

2.1.2 Object Representation

The object tracking process can have different steps to follow; object representation, object detection, object tracking, and interpretation. And to have an efficient and successful object tracker, it is necessary to have the best representation of the object of interest depending on the application we need. Usually, object representation is defined to consist of a shape representation and/or an appearance representation. A brief description of the two is given below: [4, 5].

Shape Representation

The most widely used shape representations include points, geometric shapes, silhouettes, articulated shape models, contour, and skeletal models [4, 5]. The different approaches regarding shape representation are given in Figure 2.1 below.

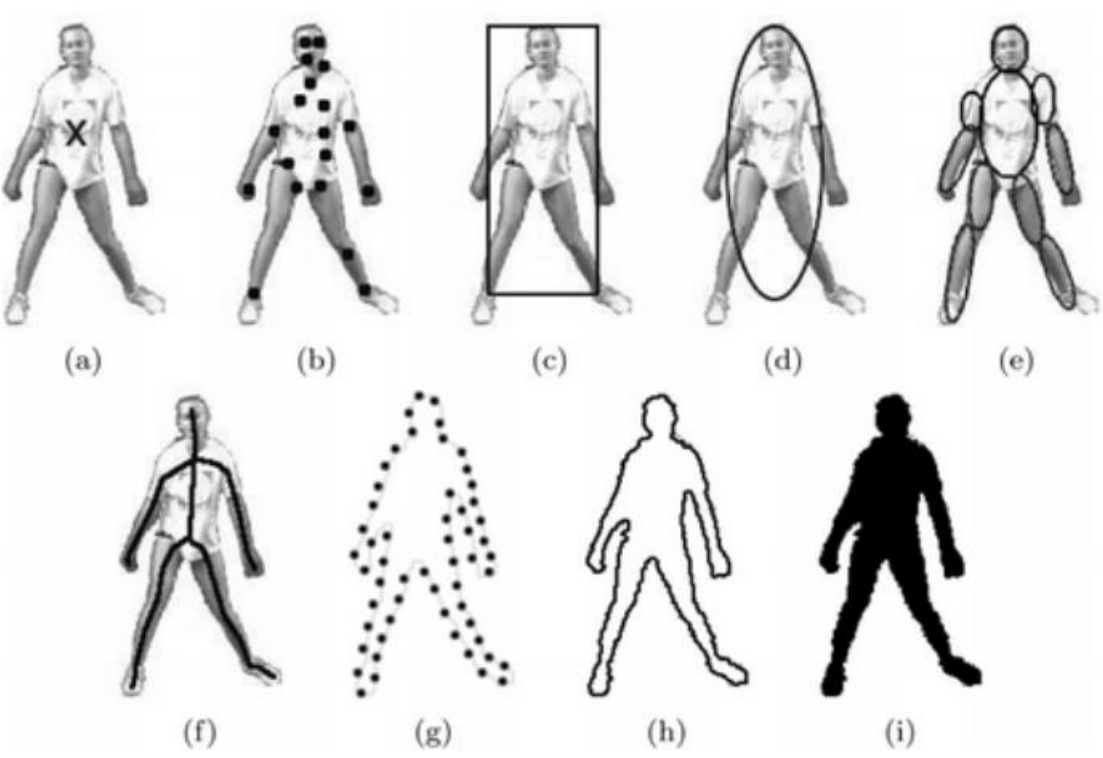


Figure 2.1: Shape representation approaches: (a) single point, (b) multi-point, (c) geometric/ rectangle, (d) geometric/ ellipse, (e) articulated shape, (f) skeletal model, (g & h) contour, (i) silhouette [4].

Appearance Representation

As for shape representation, there are various ways to represent the appearance feature of the object [4, 5]. Both shape and appearance representations are combined while doing object tracking. The most commonly used appearance representation methods are [4, 5, 8]: probability densities, templates, active appearance models, and Multiview appearance models.

2.1.3 Object Detection

Identification of stationary background from the foreground object is a very significant task but remains the most difficult problem for researchers [1]. Detecting the foreground objects is the initial step of a visual surveillance system. Object detection became more difficult with the occurrence of short and long-term dynamic scene changes. These changes include repetitive motions like waiving tree leaves, shadows, noise on the camera, light reflectance, and sudden illumination variations [1]. Hence, it is important to pay necessary attention to object detection steps to have a reliable, and fast visual surveillance system [1].

Object detection methods can be motion-based or feature-based object detectors [1]. In motion-based detection, the movement of objects can be detected as revealed by the dynamics of the scene. The most common motion-based object detection methods are frame differencing, optical flow, and background subtraction [1, 5]. A feature is a unique attribute of an object. Selecting the right feature plays a great role in tracking applications. Features include color, grey level, contour, edges, and texture [1].

After identifying the feature of the object to be represented, one can use either of the following object detection methods, which are: point detectors, background subtraction, segmentation, supervised learning, or temporal differencing [4].

2.1.4 Object Tracking

Object tracking is a problem of creating correspondence with objects or parts in a consecutive frame in order to fetch information of the object like trajectory, speed, posture, and direction, etc. [5]. It is an important part of many systems like surveillance, traffic monitoring, and air space monitoring [1, 4, 5].

There are many different algorithms that are utilized for the aim of tracking objects [4,]. Some algorithms only handle single object tracking while others explicitly handle the case of occlusion to make it possible to track multiple objects. Algorithms used for object-tracking can

be categorized into broad groups or categories. The most common way is to divide these methods into three groups which are *kernel tracking*, *point tracking*, and *silhouette tracking* [4, 5, 8]. These are usually divided into two subcategories each and are shown in Figure 2.2 below.

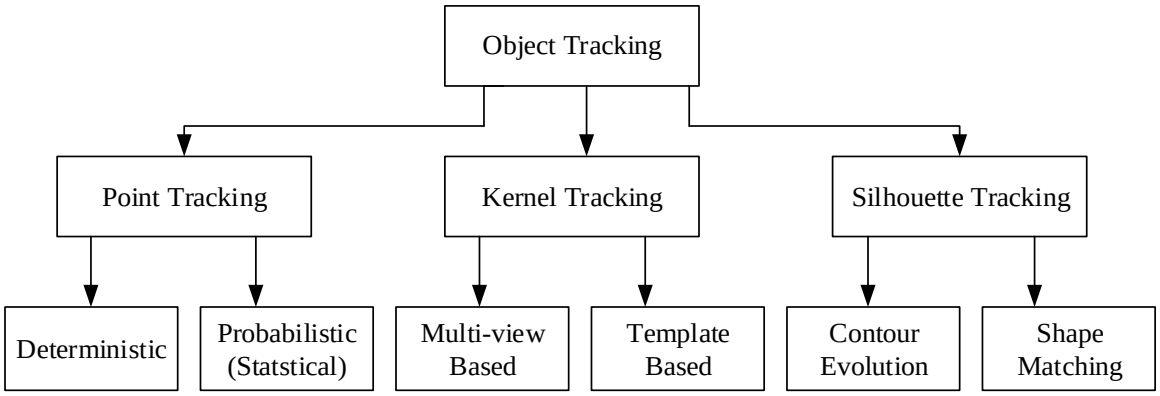


Figure 2.2: Types of object tracking approaches [4].

Point Tracking

This point tracking approach is used when one needs to denote the object of interest as point(s). Then tracking is performed by observing the state of point(s) in terms of position and motion [4]. It is made possible by associating points across frames based on the previous object states. This makes it a complicated problem especially when there has an increase of complexity like occlusion, misdetections, and entries and exit of objects [4].

Kernel Tracking

This kernel-based tracking approach is based on reckoning the movement of an object represented by a primitive object region. By employing motion models and analyzing the movement of an object from one frame to the other, determination of its next position is possible [4]. Kernel, in this view, refers to the look of the object that comprises the shape and appearance. Kernel tracking methods can be categorized into two categories as multi-view-based appearance model and template-based appearance models as shown in Figure 2.2 above [4, 5, 8].

Silhouette Tracking

Silhouette tracking is also known as region tracking or region-based tracking gives an accurate shape description of objects which have complex shapes like hands, shoulder, and head that cannot be well described by simple geometric shapes [4, 8]. The main objective of a silhouette-based object tracker is to get the object region in every frame by way of an object model

generated from the previous frame [8]. Silhouette-based object trackers could be, shape matching type or contour tracking approach. Contour tracking approaches develop an initial contour to its new position in the current frame by using state-space models or direct minimization of some energy functions while shape matching approaches to search for the object silhouette in the current frame [8].

2.2 Object Tracking in LabVIEW

Object tracking in LabVIEW is done with help of a mean-shift algorithm. A mean-shift algorithm is an effective approach for target object tracking and to acquire sequential images and is not greatly affected by the presence of similar objects [7]. In the mean-shift algorithm, the interest object's current location is searched based on the histograms of the object generated in the previous image frame. The peak of the probability density function near the object's old position is found by shifting the mean of the result [7]. The mean-shift algorithm needs the correct determination of the initial location of the target object. Having given the initial location of the target, new target locations are searched by ignoring many other similar-shaped objects as the video images progress sequentially in time [7].

2.2.1 LabVIEW

LabVIEW (Laboratory Virtual Instrumentation Engineering Workbench) is a graphical-based programing language developed by National Instruments (NI) [32, 34].

LabVIEW programs are referred to as Virtual Instruments (VIs) due to their appearance and functioning mimic physical instruments such as oscilloscopes and multi-meters [33]. Its graphical nature makes it ideal for test and measurement (T&M), automation, instrument control, data acquisition, and data analysis applications. This results in significant productivity improvement over conventional programming languages and National Instruments' focus on products for T&M, giving them a good insight into developing LabVIEW [32].

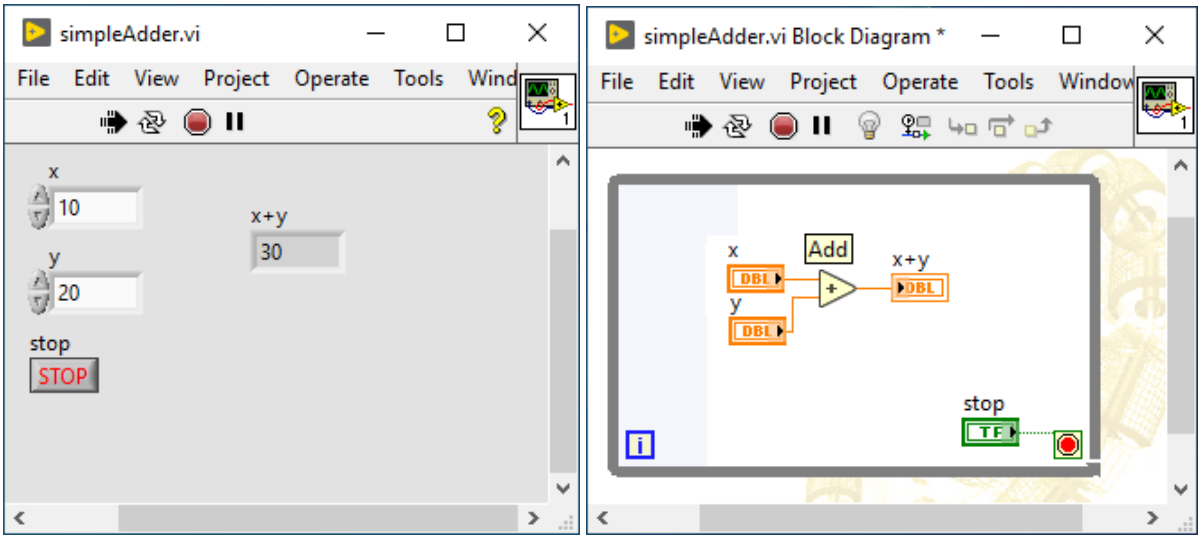
2.2.2 Virtual Instruments

Virtual Instruments (VIs) are LabVIEW programming elements that contain block diagrams, front panels, icons, and connector panes which are used to create programs [32, 33, 35]. The front panel consists of indicators and controllers while the main code of the VI is located in the block diagram. Icons are visual representations of VIs and consist of connectors for program inputs and outputs. Programing languages such as C and BASIC use functions and subroutines as a programming element, while LabVIEW uses the VI. In order to create large-scale

applications, multiple VIs ranging to several of hundreds are used. User interface elements in VIs such as graphs, controls, are drag-and-drop and this makes LabVIEW easy [32].

Front Panel

Front panels, build with controls and indicators, are the user interface part of the VI. The controls are the interactive input terminals while indicators show the output of the program for the user. Control terminals include input devices like pushbuttons, knobs, and dials while indicators show results and include graphs, LEDs, and other displaying devices [32, 35]. The front panel of the VI is shown in Figure 2.3 (a) below.



(a) (b)

Figure 2.3: Virtual Instruments: (a) front panel; (b) block diagram

Block Diagram

The objects from the front panel are controlled by the code build with a graphical illustration of functions in the block diagram [35]. The graphical source code of a program is contained in block diagrams as seen in Figure 2.3 (b) above. It contains input and output terminals, wires that transfer data between the block diagram objects, subVIs, constants, functions, and structures, etc. [35]. Connector panes and icons are also built-in LabVIEW block diagrams in order to be used in high-level VIs to reduce the complexity of the program. subVIs are also used within another VI which corresponds to a subprogram in text-based programming languages. Icons which may include texts, images, and both of the two, are graphical

representations of a VI [35]. Connector panes are used to build VIs as subVIs by setting up controls and indicators of the VI i.e., inputs and outputs of the VI [35].

2.2.3 Machine Vision Basics in LabVIEW

Machine vision (MV) is the way toward applying a scope of advancements to give an imaging-based programmed assessment, measure control, and robot direction in modern applications [7]. The essential uses for machine vision are programmed assessment and robot direction. In machine vision, PCs are let see and comprehend their current circumstance as we people do. In addition, PCs are let make moves depending on their perspective on the encompassing when required [7].

And to perform for computers vision capabilities, the following components are basic [7]: Images and Component of the imaging system.

Images

The fundamental computerized picture (image) is made out of a two-dimensional cluster of numbers. Each number in the exhibit speaks to an estimation of the littlest visual component, a pixel. The listed area of the pixel esteem in the cluster relates to the X and Y areas of the pixel inside the picture, as estimated from the upper left corner. Pictures can be partitioned into three, Grayscale picture, Binary picture, and Color picture [7].

The most ordinarily utilized picture design for finding the presence of the item, area, and size data is a binary picture [7]. The binary picture pixel has two-digit esteems, where an object has the estimation of 1 and the background has the estimation of 0 much of the time. Since there are just two qualities utilized, it is frequently called a 1-bit picture (bit depth of 1, or 2^1). To make a binary picture, the grayscale picture is generally utilized as a beginning stage. By and large, we utilize a threshold value to change over a grayscale picture to a binary picture. For the situation that the interest object in a picture is splendid against a dull background (the imaged item's pixel esteem is bigger than a picked threshold value), it is named the object (a pixel picture estimation of 1) and if the picture esteem is not exactly the threshold, it very well may be named the background (the pixel picture estimation of 0). Notwithstanding, it should be noticed that there will be situations where the dim pieces of a picture may speak to the object with the bright part including the background. When the grayscale picture is changed over to a binary picture, different picture preparation capacities can be utilized. For instance, we can

utilize the particle analysis function from which the size, zone, and focal point of the object can be handily gotten [7].

Digital color pictures from advanced cameras are normally portrayed by three-color esteems, R (red), G (green), and B (blue). The three-color esteems that speak to a picture pixel depict the color and brightness of the pixel. All in all, the brightness and color of the pixels in a picture got from an advanced color camera are commonly characterized by the blend of the R, G, and B values. All potential colors can be characterized by these three essential color mixes [7].

Component of Imaging System

The basic parts of the imaging system are the vision system (camera, lens, bus, and lighting), image/ video acquisition (computer), and image processing (software) [7].

2.2.4 Software Used for Vision Systems in LabVIEW

National Instruments offers different kinds of Vision software depending on our application and our needs. And are given below [36]:

- NI Vision Acquisition Software
- NI Vision Assistant Software
- NI Vision Development Module
- NI Vision Builder for Automated Inspections
- LabVIEW Math-Script RT Module (optional)
- LabVIEW Control Design and Simulation Module (optional)

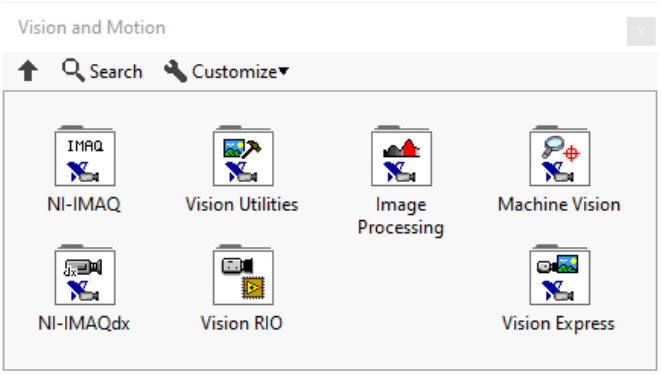


Figure 2.4: Vision and motion palette in LabVIEW

NI-Vision Acquisition Software

NI-Vision Acquisition is the fundamental programming we have, to make vision applications in LabVIEW. It incorporates vital drives, for example, NI-IMAQ and NI-IMAQdx. The NI-IMAQdx driver programming enables us to obtain pictures with IEEE 1394 (firewire), GigE Vision (Ethernet), and USB cameras [36].

NI-Vision Assistant Software

It is a device for prototyping, testing, and making picture-handling applications [7]. To model a picture preparing the application, custom algorithms with the vision assistant script component are developed. The script component records each progression of the preparing algorithm and in the wake of finishing it, one can test it on different pictures to ensure that it works [37].

NI-Vision Development Module

NI-Vision Development Module is utilized for further developed machine vision and picture preparing/processing applications. It contains many pictures handling and machine vision capacities. This bundle incorporates the underlying capacities for [36]:

- Pattern matching
- Texture recognition
- Counting and classification
- OCR (Optical Character Recognition)
- Bar code readers
- Image filters etc.

NI-Vision Builder for Automated Inspections

NI-Vision Builder for Automated Inspection (AI) is an outer and free application for building machine vision applications without the requirement for programming [36].

LabVIEW Math-Script RT Module

This module is a text-based device that is fundamentally the same as MATLAB. The syntax structure is like MATLAB, we can make and run alleged m files, and so forth. The module is accessible from the tool's menu inside LabVIEW. Math-script is an advanced, text-based programming language. Math-script incorporates in excess of 800 inherent functions and the

syntax is like MATLAB. We may likewise make custom-made m-files as we do in MATLAB [38].

In spite of the fact that Math-script is an extra module to LabVIEW, we don't have to realize LabVIEW programming to utilize it. At the point when we need to coordinate Math-script functions (built-in or custom-made m-files) as a component of a LabVIEW application and consolidate graphical and literary programming, we can work with the Math-Script Node [38].

LabVIEW Control Design and Simulation Module

The Control Design Toolkit was first dispatched in Spring 2004. It extends LabVIEW's capacities for control and dynamic system investigation and plans significantly. The arrangement of functions accessible is tantamount to the Control System Toolbox compartment in MATLAB and the comparative control system function class in Octave [39]. This module is utilized for making Control and Simulation applications with LabVIEW. Here we can discover PID regulators, Fuzzy Logic regulators, and so forth. The module is accessible as a palette on our block diagram [39].

With LabVIEW Control Design and Simulation Module we can build plant and control models utilizing transfer function, state-space, or zero-pole-gain; examine system execution with devices, for example, step response, pole-zero maps, and Bode plots; and simulate linear, nonlinear, furthermore, discrete systems with a wide choice of solvers. We can likewise dissect open-loop model behavior, design closed-loop regulators, simulate online and offline systems, and conduct physical implementations [39].

2.3 Controllers

2.3.1 PID Controller

The PID (Proportional-Integral-Derivative) controller algorithm is among regularly utilized control algorithms in process industries (like heating and cooling systems, flow control, pressure control, and fluid level monitoring, and so on) because of its simplicity, and give an acceptable performance [12]. While using the PID controller, one need to indicate a process variable (system parameters we need to control such as temperature, pressure, or flow rate) and a set point (a desired value for the above parameters we will control). A PID controller generates a control signal and applies it to the system which thusly will drive the process to its desired value [12].

PID controllers give better execution just at a specific working range and their performance diminishes as the working boundaries change and should be re-tuned when the working range let changed. In PID controller structure, we have four types of controllers that belong to the family of PID controller and these are P (Proportional Controller), PI (Proportional plus Integral Controller), PD (Proportional plus Derivative Controller), and PID (Proportional plus Integral plus Derivative Controller) [30].

2.3.1.1 PID Algorithms

In PID controller, the controller generates the control signal (controller action), u(t), from the error signal, which is the contrast between the ideal yield and the deliberate yield of the system, as seen in the equation below [12]:

$$u(t) = K_c\left(e + \frac{1}{T_i}\int_0^t e\,dt\right) + T_d\frac{de}{dt} \tag{2.1}$$

Where K_c is controller gain and error, e, is the difference between desired output and the actual output of the system. T_i and T_d are the integral and derivative control gains respectively. The Laplace Transform of equation (2.1) above is:

$$U(s) = K_c(1 + \frac{1}{T_i s} + T_d s)E(s) \tag{2.2}$$

In terms of error: the *proportional* controller part depends on the present error signal; the *integral* controller part depends on the accumulation of past error signals, and the *derivative* controller part forecasts the future error signal.

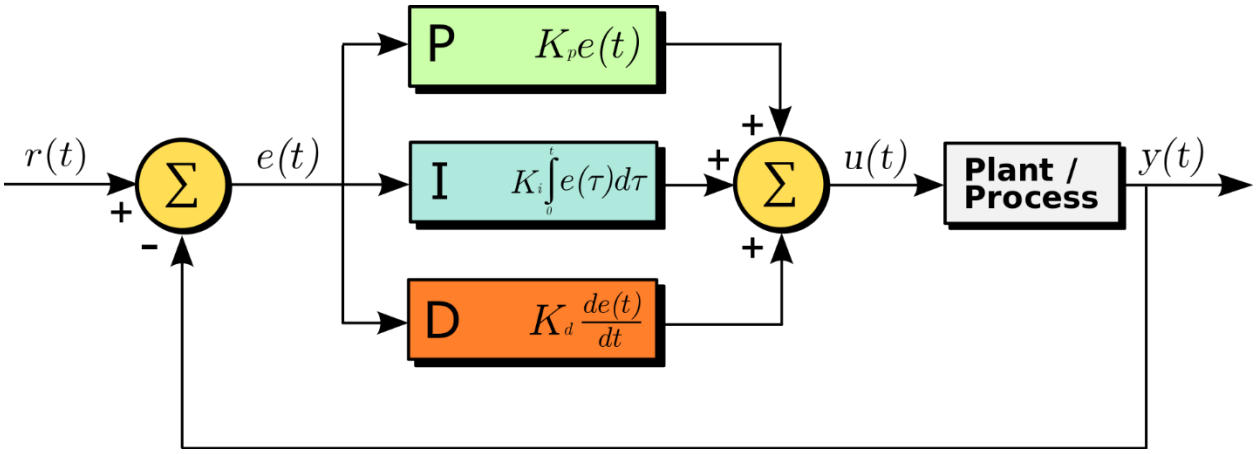


Figure 2.5: PID controller structure [40].

2.3.1.2 Tuning Mechanism for PID Controllers

Tuning is a method of choosing the PID controller parameters to meet the necessary plant execution specifications. Many tuning approaches were created for PID controllers and can be named the manual and automatic tuning strategies. Manual tuning methods are comparatively inefficient especially when the computation time becomes longer [31].

The manual tuning method here is based on the work of Ziegler and Nichols, the developers of the quarter-decay ratio tuning techniques derived from a combination of theory and empirical observations [12]. The closed-loop quarter decay ratio values are given in Table 2.1 below. The procedure used in this tuning method is as given in [12].

Table 2.1: Closed-loop quarter decay ratio values [12].

Controller	PB (Percent)	Reset (Minutes)	Rate (Minutes)
P	$2.00PB_u$	---	---
PI	$2.22PB_u$	$0.83T_u$	---
PID	$1.67PB_u$	$0.50T_u$	$0.125T_u$

2.3.2 Fuzzy Logic Controller

Modern control systems theory has arisen during the Second World War, when the plan, examination, and synthesis of servomechanisms were fundamental in the assembling of electromechanical systems [10]. The advancement of control systems theory has since experienced a developmental process, beginning from some essential, oversimplified, frequency-domain examination for single-input single-output (SISO) linear control systems, and summed up to a numerically refined modern theory of multi-input multi-output (MIMO) linear or nonlinear systems portrayed by differential and/or difference equations [10].

The improvement of space innovation during the 1950s totally changed the soul and direction of the traditional control systems theory: the difficulties presented by the high accuracy and extraordinary unpredictability of the space systems [10].

A computer scientist at the University of California Berkeley, Lofty Zadeh proposed the fuzzy set theory in 1965. "A fuzzy set is a class of objects with a continuum of grades of membership." Lofty characterized fuzzy sets as a class of sets with grades of membership from 0 to 1. Advanced Systems, Artificial Neural Networks, and Fuzzy Systems share the property

of being model-free approximations, which implies that no accurate mathematical model of the actual system to control or to estimated is required [11].

The point of fuzzy control systems theory is to expand the existing fruitful conventional control systems procedures and strategies as effective as possible and to create numerous new and extraordinary purposed ones, for a lot bigger class of complex, complicated, and badly modeled systems – fuzzy systems and developed manly to solve real-world problems [10].

Fuzzy controllers are utilized to control fuzzy systems. Most conventional control algorithms require the mathematical model of the system to be controlled and numerous actual systems are sufficiently complex to locate their mathematical models [12]. Likewise, numerous processes are either non-linear or hard to be controlled with the conventional control algorithms. And hence a qualitative portrayal or description of the control system is a superior fit for such complex processes [12].

2.3.2.1 Fuzzy Logic Concepts

Fuzzy sets plan to model the uncertainty or ambiguity related to natural human reasoning, which depends on linguistic words and sentences rather than in mathematical expressions and relations. It depends on approximate reasoning (or fuzzy reasoning) which is a reasoning method of neither exact nor inexact and it is basic for fuzzy inference system. To see how it is performed, three fundamental ideas should be characterized [11]:

- Linguistic Variable: is a variable whose qualities are words or sentences in a natural or artificial language, not numbers. For example, the variable temperature can be portrayed as it is introduced underneath.

$$\text{Temperature} = \{\text{Cold, Moderate, Hot}\} = \{\text{C, M, H}\} \quad (2.3)$$

- Fuzzy Proposition: is a statement expressed in a natural or artificial language. Rather than classical logic suggestions, a fuzzy proposition may receive a truth-value from the interval [0, 1]. For instance, the Temperature is Hot.
- Linguistic Rule: is an IF-THEN rule utilized for setting the activities that should be possible by the controller. It has two sections: Antecedent Part (premise), expressed by: if < fuzzy proposition>, and Consequent Part, expressed by: then < fuzzy proposition>.

A fuzzy set, *F*, is characterized by a membership function which assigns to each element a grade of membership and is entirely defined by the set of ordered pairs.

$$F = \{(x, \mu_F(x)) \mid X \epsilon U\} \text{ and } \mu_F: U \rightarrow [0, 1], \quad (2.4)$$

Where x is an element of the universe of discourse U and μ_F is a membership function that assigns a degree of membership $\mu_F\ (x)$ to each element x of F.

Inputs in a Fuzzy logic system can get membership values between 0 and 1, but in conventional logic, the membership value is limited to only two values, 0 or 1 [11].

Membership Functions

Membership functions are numerical functions that relating to linguistic terms. It speaks to the level of membership of linguistic values inside their linguistic terms. The degree of membership is nonstop somewhere in the range of 0 and 1, where 0 is equivalent to 0% membership and 1 is equivalent to 100% membership [11]. A fuzzy set is totally described by its *MF* (membership function) indicated by μ and it tends to be a discrete or a continuous universe of discourse. The typical continuous universe of discourse shapes of membership functions is: trapezoidal, triangular, Gaussian, sigmoidal, generalized bell and so forth.

2.3.2.2 Fuzzy Logic Control Components

The basic fuzzy logic control components consist of: fuzzification, rule base, decision making (computation) and defuzzification [31].

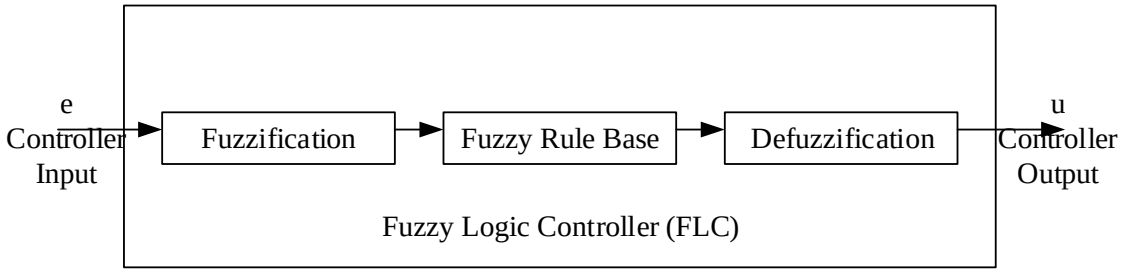


Figure 2.6: General structure of a Fuzzy Logic Controller [10].

Fuzzification Methods

Fuzzification is the way of associating crisp or numerical input values with the linguistic terms of the corresponding input linguistic variable [12]. It is the way of representation of a crisp value by a membership function. For example, we can have the accompanying tasks: the room temperature of 50 degrees can associate with a linguistic term of Cold with a degree of membership 0.4, the room temperature of 70 degrees can relate to a linguistic term of Moderate with a degree of membership 0.8, the room temperature of 100 degrees can relate to a linguistic term of Hot with a degree of membership 1 and so forth [12].

Fuzzy Rule Base

The rule base is a critical component of fuzzy logic to make the controller work appropriately and viably [10]. Its responsibility is to create the control activity, in fuzzy terms, in light of the data gave by the fuzzification module and control application prerequisites. All the more explicitly, the rule base is a bunch of IF-THEN rules which has the accompanying structures [10]:

R^1: IF controller input e_1 is E_{11} AND ... AND controller input e_n is E_{1n}, THEN controller output u_1 is U_1.

.

.

.

R^m: IF controller input e_1 is E_{m1} AND ... AND controller input e_n is E_{mn}, THEN controller output u_m is U_m.

The foundation of this rule base relies intensely upon the designer's work experience, information about the actual plant, analysis and design skills, and so forth, and is, consequently, more or less subjective [10].

Defuzzification Methods

Defuzzification is the process of changing the degrees of membership of output linguistic variables within their linguistic terms into crisp numerical values [12]. Fuzzy regulators utilize an implication strategy to scale the membership function of output linguistic variables prior to performing defuzzification. Among a few numerical techniques used to perform defuzzification are Center of Area (CoA), modified Center of Area (mCoA), Center of Sums (CoS), Center of Maximum (CoM) and Maximum of Maximum (MoM) [10, 12].

2.4 Genetic Algorithm

Genetic Algorithm (GA), which utilizes the idea of Darwin's theory of evolution, is a search heuristic. It attempts to play out an intelligent search for an optimal or approximate solution from a boundless number of potential solutions. The algorithm utilizes a process of natural selection where the fittest individuals are chosen to create offspring of the next generation [42, 43].

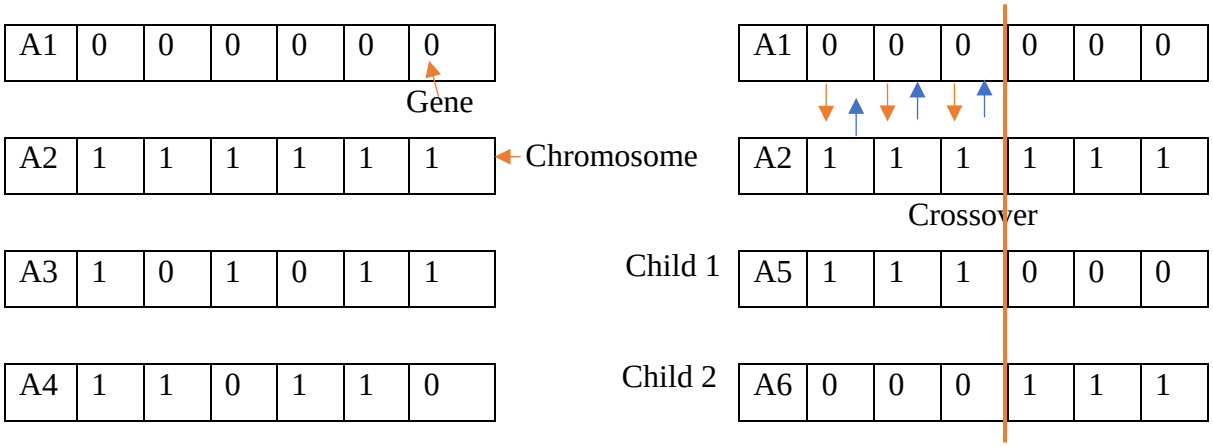


Figure 2.7: Genetic algorithm crossover operation [43].

Notion of Natural Selection

The process of natural selection begins from the distinguishing proof of fittest individuals from the available populaces [43]. At that point, they produce children which acquire the attributes of the parents and are utilized for the reproduction of the next generation. The chance of survival of the newly produced offspring will be better than parents if the parents have better fitness. The process keeps iterating until a generation with the fittest individual is found.

In genetic algorithm, the following five phases are considered:

- Initial population size
- Fitness function
- Selection
- Crossover and
- Mutation

In a genetic algorithm, an initial population size of n strings for n parameters of length L is created. The strings are created in a random fashion by placing the zeros and ones in the string arbitrarily. The strings at that point are decoded into a bunch of boundaries that it speaks to. The set of parameters is passed by a mathematical model of the problem space. Depending on the input set of parameters the mathematical model gives out a solution. On the basis of the efficiency and character of the solution, the string is allotted a *fitness* value. The *fitness* values are evaluated for each string in the entire population. Using these fitness values, a new generation of strings are produced which is expected to perform better than their parents [43].

Then again, individuals with the best fitness value are *selected* and let give their genes to the next generation. The selected individuals are then mated by choosing a *crossover* point randomly from within the genes. Crossover is the most significant phase of the genetic algorithm. While producing new offspring, there is a probability that some of their genes can be subjected to a *mutation* with a very low tendency to occur i.e., some of the bits in the bit string can be flipped. It happens to keep up variety inside a populace and forestall untimely combination. The algorithm then terminates when the population converges (does not produce offspring which are significantly different from the previous generation) and gives a set of solutions to our problem [43].

The algorithm mostly ends when either an acceptable fitness level has been reached, or a maximum number of generations has been produced. When the algorithm is terminated due to a satisfactory fitness level then the satisfactory solution is expected, but if it is terminated due to a maximum number of generations then the required solution may or may not have been reached.

2.5 Literature Review of Related Recent Researches

Due to their wide range of applications, object tracking techniques have become a rapidly growing research area. A summary of works of literature published regarding the object-tracking system is given next. Considerable care is taken to recognize and summarize related and publicly available research papers.

The author in [13] addressed image tracking techniques in LabVIEW and tried to detect, recognize and track the circular-shaped image among the other shapes. The researcher generates the path of the image on the script module. But he did not suggest the type of object tracker he designed with respect to the field-of-view of the camera i.e., fixed or dynamic background. As the simulation results put on indicates the camera he has used was fixed (static) and cannot follow moving objects out of the field-of-view of the camera.

Another set of authors in [2] used a mean-shift algorithm in order to follow objects from a video source. The video object is recorded and stored as an AVI file, then they tried to follow the object of interest from the video. They did not use live video capturing cameras and follow the object of interest in real-time.

The researchers in [23] had investigated a real-time eye-tracking algorithm using a NI-smart camera. They had captured and tracked the eye with NI smart camera, LabVIEW and NI Vision Builder. They had simulated their system with a mounted camera and a fixed distance to the object. The eye-tracking accuracy decreases with increasing frame rates, which shows that no control algorithm has been implemented.

The authors in [46] have proposed a smart autonomous camera tracking system with a pan/tilt mechanism. They tried to track the object based on its color and pan/tilt adjustment is done in order to automatically adjust the region of interest captured by the camera. NI myRIO module is used to generate the appropriate pulse width modulation (PWM) signals for the servo motors used as a pan/tilt mechanism. But in this research, no way of servo motor position control techniques has been suggested.

The other set of researchers [9] used a Neuro-Fuzzy based approach for motion prediction and tracking. The interest object tracking system model was developed with the help of a Neuro-Fuzzy hybrid-based approach to predict the object's trajectory and follow its path. The Neuro-Fuzzy hybrid approach was used to design the fuzzy rule base of the intelligent system. They have used ANFIS methodology in order to build a Segeno fuzzy model for controlling the position of a servo motor while carrying the charge-coupled device camera (CCD). They finally had obtained the time responses of the azimuth (pan) and elevation (tilt) mechanism as rise time, $t_r = 0.1$ and $0.3\ seconds$ respectively. The settling time $t_s = 0.5\ second$ for both and zero steady-state error has been reported with the ANFIS controller [9]. But they have not suggested any tracking algorithm used for the moving object detection.

Chapter Three

Methodology

3.1 Introduction

To follow the object of interest, the camera is set to track the object by adjusting the x and y coordinates automatically as it moves. In other words, from the camera's viewpoint, the object is always set to remain at the center of the camera. The camera is constantly directed towards the center of the object of interest.

While tracking an object, if the interest object is not at the center point of the camera, the error signals are generated that show the deviation from the object center to the camera center. Based on these error signals a correcting measure is taken by the controller. In Figure 3.1 shown below, the camera center is defined as (320, 240) and the tracked object center location is given by (a, b) in the frame. Here the resolution of each frame from the camera is assumed to be (640, 480), hence the pixel point (320, 240) is the center point of the camera.

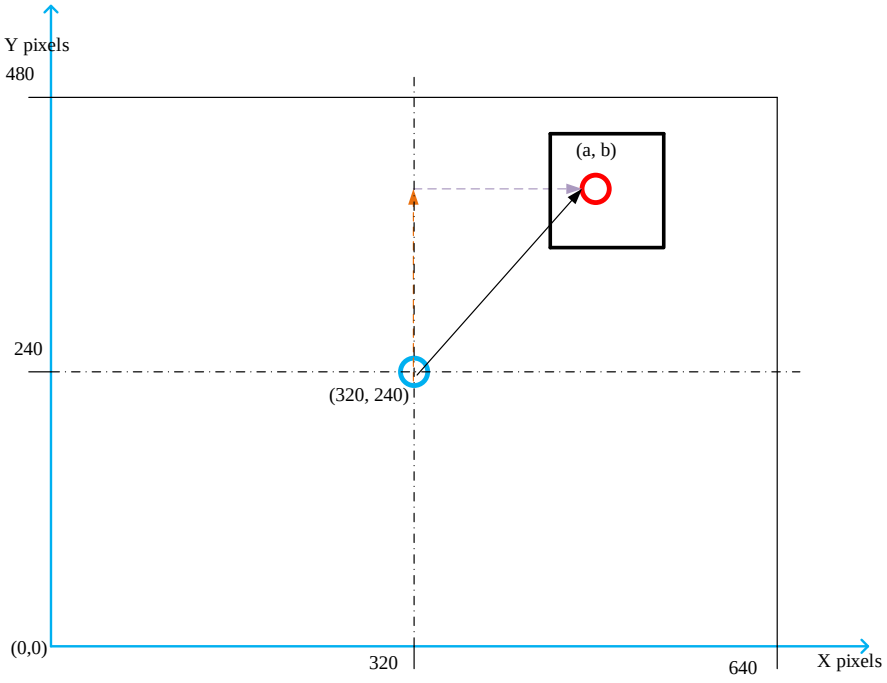


Figure 3.1: Coordinate system from captured frame

The distance error signal is defined as the difference between the camera center coordinate and object center coordinate. The x and y error signals' deviation from the center point of the camera to the object center are given respectively in the following equations.

$$x_{error} = \frac{(320 - a)}{640} \tag{3.1}$$

$$y_{error} = \frac{(240 - b)}{480} \tag{3.2}$$

Where x_{error} represents the horizontal error obtained due to the camera's horizontal movement and y_{error} represents the vertical error obtained when the camera moves in a vertical plane in order to stick with the object.

In addition, the size of the interest object from the camera's perspective becomes smaller when the distance measured between the camera and the interest object increases, and it becomes larger when the distance measured between the two decreases. And this generates the area error signal as of the distance error signal above. The bounding box's size, created to represent the object of interest, changes as the object approaches or departs the camera. The area error signal can be calculated as in equation (3.3) below. From this area error signal, one can identify whether the object is approaching or departing the camera.

$$A_{error} = \frac{A_{initial} - A_{current}}{A_{initial}} \tag{3.3}$$

Where A_{error}, $A_{initial}$, and $A_{current}$ are the area error, initial area of the bounding-box and current area of the bounding-box as the object approaches or departs the camera respectively.

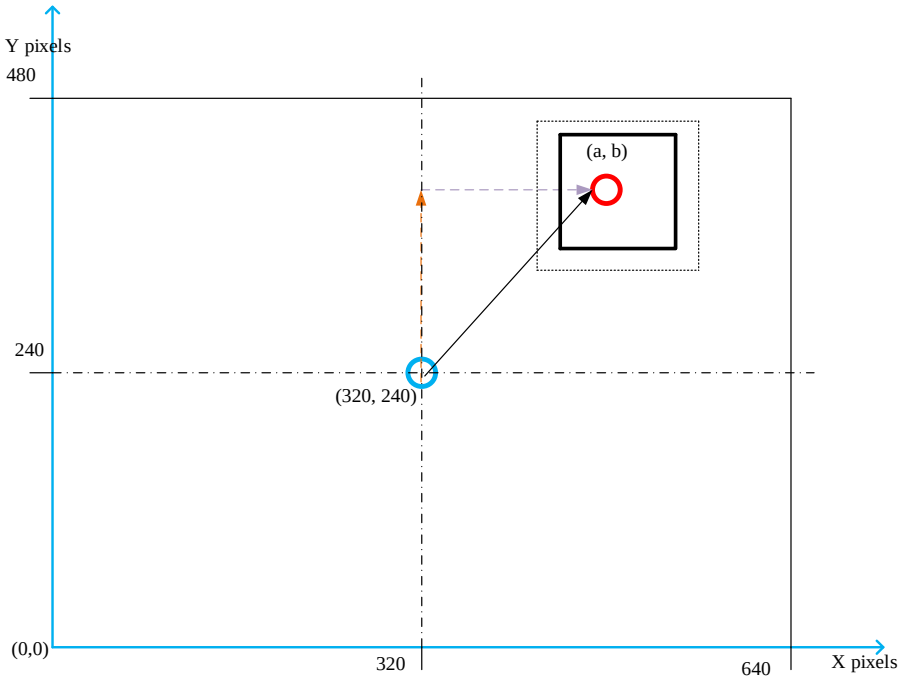


Figure 3.2: Coordinate system when object gets far from the camera

Object tracking is then achieved by using these error signals as feedback and design controllers in order to minimize the errors and let the camera always follow the interest object in desired direction and speed.

In perfect tracking, the object center (a, b) aligns with the camera center (320, 240) and in this case, both the horizontal and vertical errors become zero. The area error can be used to keep the distance between the camera and the object if the camera is let free to move in the z-direction, but this is not the concern of this work.

3.2 System Description

The proposed object tracking system comprises a camera for object vision, two servo motors to perform high accuracy positioning system (pan and tilt mechanism). It additionally has a regulator to efficiently follow the object under the scene by changing the position of the servo motors with a feedback system. The camera is mounted to the pan motor shaft, and the tilt motor is attached in a manner to give a tilt move for the camera and pan motor configuration.

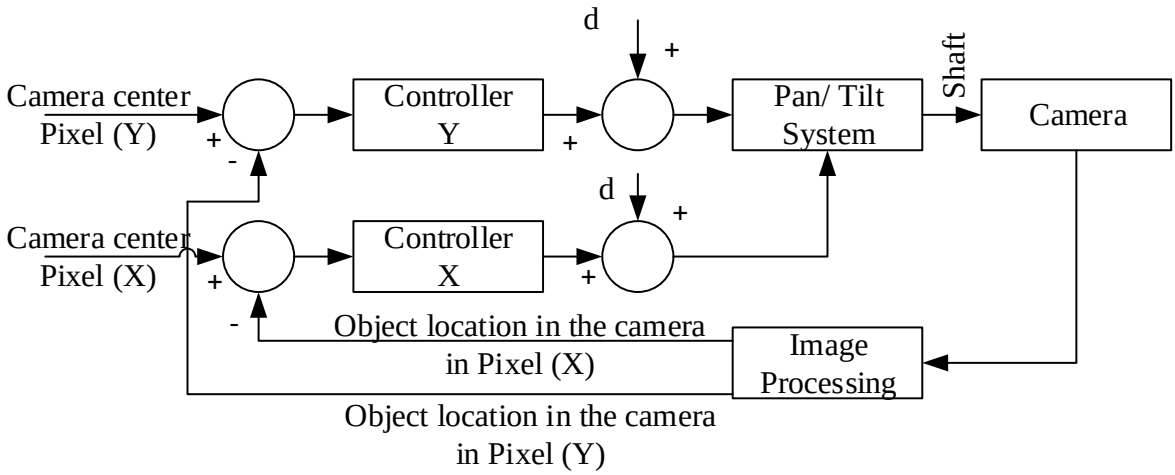


Figure 3.3: Block diagram of object tracking system connections.

3.3 Pan/ Tilt Mechanism Model

The pan and tilt mechanism of the tracking system can be modeled as two separate actuators. The one actuator corresponds to the pan movement, and the other one corresponds to the tilt movement of the camera. The camera here acts as a sensor. It captures the interest object with the help of NI-Vision and after some image processing techniques, object's x and y location on the captured frame are extracted. These object locations in a 2-D plane are then feedbacked to the controller in order to control the position of the actuators efficiently. The actuators used

for this purpose are servo motors. Servo motors are electromechanical devices used to push or rotate interest objects with greater precision to some specific angles or distance. They are constructed with a simple motor that operates through a servomechanism. They can be DC servo motors or AC servo motors based on the used power source [41].

3.3.1 Mathematical Modeling

The type of servomotor used in this thesis work is separately excited armature-controlled DC servomotor. It is selected as it is the most well-known and common motor type and exhibits excellent torque-speed characteristics and can be controlled by changing the voltage signal associated with the input [9]. Likewise, armature-controlled DC servomotor is a closed-loop system that let it favored over field-controlled DC servomotor that is an open-loop system. The equivalent circuit of this motor is given in Figure 3.4 beneath [9].

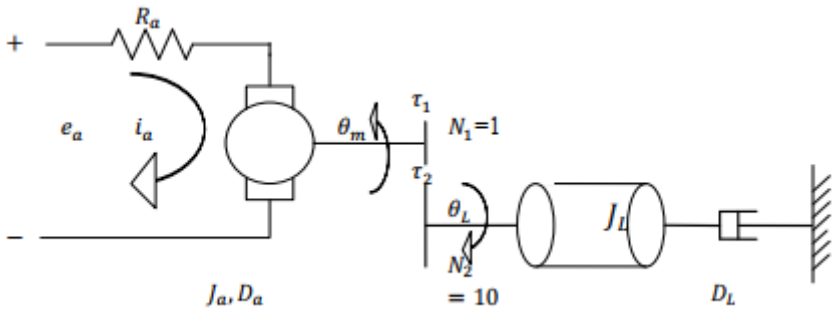


Figure 3.4: The equivalent circuit representation of DC servomotor and a gear load [9].

As the motor is in armature-controlled mode, the field current is kept constant. Then the torque on the motor T_M is related to the armature current I_a by a constant factor and is given by,

$$T_M = K_T I_a \tag{3.4}$$

Where K_T is the given motor torque constant in (N-m/A) and I_a is the armature current in (A). Similarly, the back EMF E_b is related to the angular velocity ω as follows,

$$E_b = K_b \omega = K_b \frac{d\theta_m}{dt} \tag{3.5}$$

Where K_b is back EMF constant in (volt-sec/rad) and θ_m is angular displacement at the motor shaft.

DC servomotors have both electrical and mechanical parts. The electrical part comprises resistance, inductance, input voltage, and the back EMF while the mechanical part comprises

a motor shaft, inertia of the motor, and load and damping. With the help of Kirchhoff's and Newton's laws, the non-linear differential equation of a DC servo motor is given beneath [9].

$$L_a \frac{di_a}{dt} + R_a I_a = V_a - K_b \omega \tag{3.6}$$

$$J \frac{d\omega}{dt} + D\,\omega = K_T I_a - T_w \tag{3.7}$$

Where V_a, T_w, J and D are the applied armature voltage in volts, the disturbance torque in Newton-meter (N-m), Moment of Inertia (Kg-m^2) and Viscous-friction coefficient (N-m/KRPM) respectively. The Laplace Transform of Equation (3.6) is given by:

$$L_a s\, I_a(s) + R_a I_a(s) = V_a(s) - E_b(s) \tag{3.8}$$

$$(L_a s + R_a) I_a(s) = V_a(s) - E_b(s) \tag{3.9}$$

Transfer function of the process involved in the primary loop relating $I_a(s)$ and $V_a(s) - E_b(s)$ is given by:

$$\frac{I_a(s)}{V_a(s) - E_b(s)} = \frac{1}{(L_a s + R_a)} \tag{3.10}$$

Similarly, the Laplace Transform of Equation (3.7) setting T_w = 0 is given by:

$$J\, s\, \omega(s) + D\, \omega(s) = K_T I_a(s) - 0 \tag{3.11}$$

$$I_a(s) = \frac{(Js + D)\omega(s)}{K_T} \tag{3.12}$$

Transfer function of the process that involved in the primary loop relating ω(s) and $I_a(s)$ is given by:

$$\frac{\omega(s)}{I_a(s)} = \frac{K_T}{(Js + D)} \tag{3.13}$$

The non-linear differential equations of the DC servomotor given in Equations (3.6) and (3.7) are used for multi-objective cascade-control system design of DC servomotor, thus maintaining the non-linear characteristics [9].

In our case, the DC motor shaft is attached to a gearbox with ratio ($K_g = \frac{N_l}{N_m}$), (where N_l and N_m are the number of teeth on the load side and the number of teeth on the motor side respectively), and then the camera is attached to the output shaft of the gearbox. The gear ratio relates motor shaft angular position θ_m to the load shaft angular position θ_l as $K_g = \frac{\theta_m}{\theta_l}$). The

load inertia J_l acting at the output shaft of the gearbox reflected to the motor shaft side is given by $\frac{1}{{K_g}^2}J_l$. As torque is utilized to drive the motor shaft and the camera load attached, the torque equation in Equation (3.7) above can be rewritten as below assuming no disturbance torque [9]:

$$T_M = K_T I_a = J_m \frac{d^2\theta_m}{dt^2} + \frac{1}{{K_g}^2} J_l \frac{d^2\theta_m}{dt^2} + \frac{1}{{K_g}^2} D \frac{d\theta_m}{dt} \tag{3.14}$$

$$T_M K_g = J_{eq} \frac{d^2\theta_l}{dt^2} + D \frac{d\theta_l}{dt} \tag{3.15}$$

Where, $J_{eq} = J_m {K_g}^2 + J_l$, is the total load inertia reflected at the motor shaft and D is the rotational viscous friction constant. Taking the Laplace Transform of Equation (3.14), the below equation is obtained:

$$T_M(s) = J_m s^2 \theta_m(s) + \frac{1}{{K_g}^2} J_l s^2 \theta_m(s) + \frac{1}{{K_g}^2} D s \theta_m(s) \tag{3.16}$$

After having some algebraic manipulations, the equations relating $\theta_m(s)$ with $V_a(\mathrm{s})$ and $\theta_l(s)$ with $V_a(\mathrm{s})$ are given below:

$$\frac{\theta_m(s)}{V_a(\mathrm{s})} = \frac{{K_g}^2 K_T}{L_a J_{eq} s^3 + \left(L_a D + R_a J_{eq}\right) s^2 + (R_a D + {K_g}^2 K_T K_b) s} = T \tag{3.17}$$

$$\frac{\theta_l(s)}{V_a(\mathrm{s})} = \frac{K_g K_T}{L_a J_{eq} s^3 + \left(L_a D + R_a J_{eq}\right) s^2 + (R_a D + {K_g}^2 K_T K_b) s} \tag{3.18}$$

The block diagram representation of the motor and the load together with the gear is presented in Figure 3.5 below.

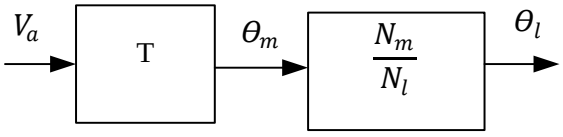


Figure 3.5: Representation of the transfer function between load angle (angle at secondary gear) and armature voltage.

The transfer function from the input voltage $V_a(\mathrm{s})$ to the load shaft angular velocity $\omega_l(\mathrm{s})$ is given by:

$$\frac{\omega_l(s)}{V_a(s)} = \frac{K_g K_T}{L_a J_{eq} s^2 + (L_a D + R_a J_{eq}) s + (R_a D + K_g{}^2 K_T K_b)} \quad (3.19)$$

As the characteristic equation is a second order, take two simple real roots as r_E and r_M. Applying partial fraction expansion to Equation (3.19) above the below equation is obtained:

$$\frac{\omega_l(s)}{V_a(s)} = \frac{A_E}{s + r_E} + \frac{A_M}{s + r_M} \quad (3.20)$$

The forced response of a system to input $V_a(t)$ (with zero initial conditions) after applying the Laplace Transform is given by:

$$\omega_l(t) = \int_0^t [A_E e^{-r_E(t-q)} + A_M e^{-r_M(t-q)}] V_a(q) dq \quad (3.21)$$

In most practical armature-controlled DC servomotor applications, $r_E >> r_M$ and this lets the electrical subsystem respond well faster than the mechanical subsystem. Due to this, the first exponential term in Equation (3.20) above decays rapidly which leads the response $\omega_l(t)$ to be dominated only by the mechanical subsystem $\frac{A_M}{s+r_M}$. In other words, the effect of the electrical subsystem component which is, $\frac{A_E}{s+r_E}$, on the response $\omega_l(t)$ in most DC servomotor control applications is neglected. This is effective by neglecting the effect of the armature inductance, L_a. And using this simplification, the DC motor load angular velocity response, $\omega_l(s)$, to the armature voltage input, $V_a(s)$, transfer function given in Equation (3.19) above can be approximated by a first-order transfer function model, and is given below [9]:

$$\frac{\omega_l(s)}{V_a(s)} = \frac{K_g K_T}{J_{eq} R_a s + (D R_a + K_g{}^2 K_T K_b)} \quad (3.22)$$

In SI units K_T *and* K_b numerical values are equal, and the transfer function in Equation (3.22) above can be reduced to:

$$\frac{\omega_l(s)}{V_a(s)} = \frac{K}{\tau s + 1} \quad (3.23)$$

Where K and τ are DC gain and mechanical time constant of the DC servomotor respectively.

3.3.2 Closed-loop Position Control of Servo Motors

The new transfer function, obtained from Equation (3.20) by neglecting the electrical component, relating the output shaft angular position to the input voltage is given below:

$$\frac{\theta_l(s)}{V_a(s)} = \frac{A_M}{s(s+r_M)} \tag{3.24}$$

The s- domain unit step response of the output angular position is:

$$\theta_l(s) = \frac{A_M}{s(s+r_M)}\frac{1}{s} \tag{3.25}$$

The final value of the response by employing final value theorem is given by:

$$\lim_{t\to\infty}\theta_o(t) = \lim_{s\to 0} s\theta_l(s) = \lim_{s\to 0} s\left(\frac{A_M}{s(s+r_M)}\frac{1}{s}\right) = \infty \tag{3.26}$$

As seen from Equation (3.26) above, the final value of the response is unbounded and hence in order to control the output position follow the input position command, the output position is feedback to the input and this insures it. Considering only a proportional control, the equation for the control unit is given below [9, 16]:

$$V_a(s) = K_p[\theta_i(s) - \theta_l(s)] \tag{3.27}$$

From Equation (3.24) above we have:

$$V_a(s) = \frac{s(s+r_M)}{A_M}\theta_l(s) \tag{3.28}$$

Then the block diagram showing the servo motor position control using position feedback is given below.

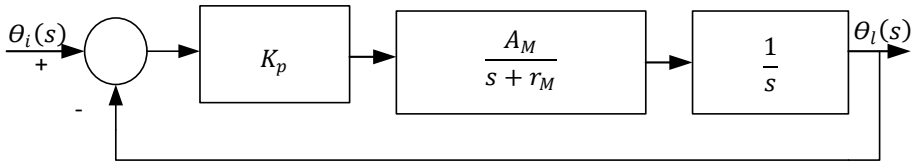


Figure 3.6: Servo motor position control using position feedback.

The transfer function relating $\theta_l(s)$ to $\theta_i(s)$ is given as follows:

$$\frac{\theta_l(s)}{\theta_i(s)} = \frac{A_M K_p}{s^2 + r_M s + A_M K_p} \tag{3.29}$$

Equating Equations (3.20) and (3.22) above and with some rearrangement A_M and r_M can be given as:

$$A_M = \frac{\eta_g K_g K_T}{R_a J_{eq}} \tag{3.30}$$

$$r_M = \frac{DR_a + \eta_g K_g{}^2 K_T K_b}{R_a J_{eq}} \tag{3.31}$$

Where, η_g, is the motor gear box efficiency.

Using parameters obtained from servo motor vendors [44], the specifications of DC servo motors employed in the modeling process are presented in Table 3.1 below:

Table 3.1: Specifications of DC servo motor [44]

Parameter	Definition	y-motor (elevation)	x-motor (azimuth)
P_R	Rated output power (W)	80	60
V_R	Rated armature voltage (V)	24	24
T_R	Rated torque (Nm)	0.26	0.19
I_R	Rated armature current (A)	5.0	3.9
N_R	Rated speed (rpm)	3000	3000
T_s	Continuous stall torque (Nm)	0.32	0.24
T_P	Peak stall torque (Nm)	2.16	1.8
I_s	Armature stall current (A)	5.2	4.5
I_p	Peak armature stall current (A)	40	31
Q_R	Rated power rate (kW/s)	1.8	1.6
K_T	Torque constant (Nm/A)	0.06	0.057
K_b	Back EMF constant (Vs/rad)	0.66 x 10^{-3}	0.63 x 10^{-3}
J_{eq}	Rotor inertia (kg m^2)	0.37 x 10^{-4}	0.22 x 10^{-4}
D	Damping constant (Nms/rad)	8.22x 10^{-3}	2.97x 10^{-3}
R_a	Armature winding resistance (Ω)	0.44	1.1
L_a	Armature inductance (mH)	0.3	0.5
τ_m	Mechanical time constant (ms)	4.5	7.4
τ_e	Electrical time constant (ms)	0.61	0.45
K_g	Gear ratio	0.1	0.1
η_g	Motor efficiency	0.87	0.87

The elevation and azimuth actuator transfer functions are then obtained by employing Equations (3.30) and (3.31) as follows:

For x-motor (azimuth):

$$A_M = \frac{\eta_g K_g K_T}{R_a J_{eq}} = 204.92 \tag{3.32}$$

$$r_M = \frac{DR_a + \eta_g {K_g}^2 K_T K_b}{R_a J_{eq}} = 135.01 \tag{3.33}$$

$$\frac{\theta_l(s)}{\theta_i(s)} = \frac{204.92K_p}{s^2 + 135.01s + 204.926K_p} = \frac{204.92}{s^2 + 135.01s + 204.92} \tag{3.34}$$

For y-motor (elevation):

$$A_M = \frac{\eta_g K_g K_T}{R_a J_{eq}} = 320.64 \tag{3.35}$$

$$r_M = \frac{DR_a + \eta_g {K_g}^2 K_T K_b}{R_a J_{eq}} = 222.18 \tag{3.36}$$

$$\frac{\theta_l(s)}{\theta_i(s)} = \frac{320.64K_p}{s^2 + 222.18s + 320.64K_p} = \frac{320.64}{s^2 + 222.18s + 320.64} \tag{3.37}$$

Having obtained the Equations (3.34) and (3.37) above, the unit step response of the modeled pan/tilt mechanism is simulated and given in the below figure.

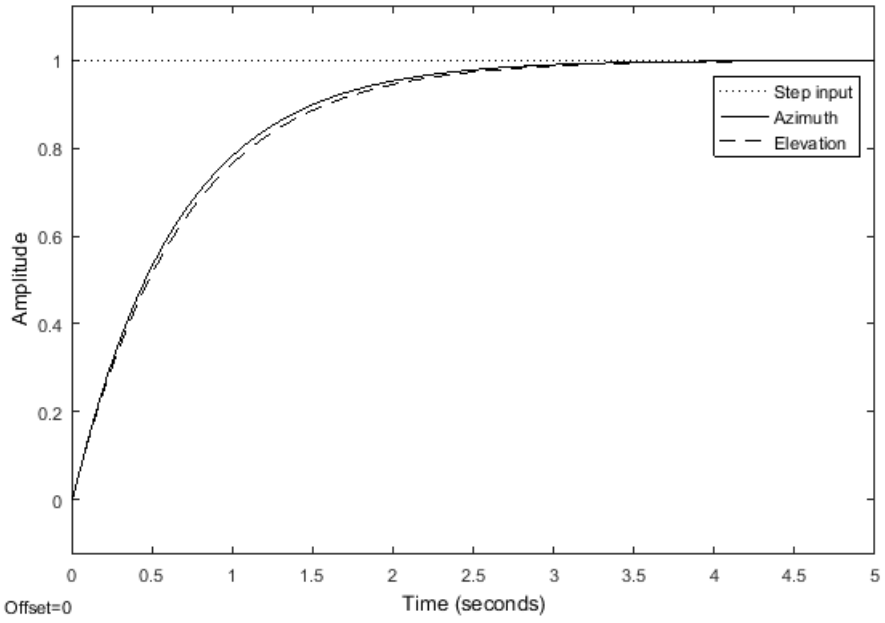


Figure 3.7: Step response of the actuators.

3.4 Methodology of Object Tracking in LabVIEW

The method used while doing object tracking techniques in LabVIEW is given below.

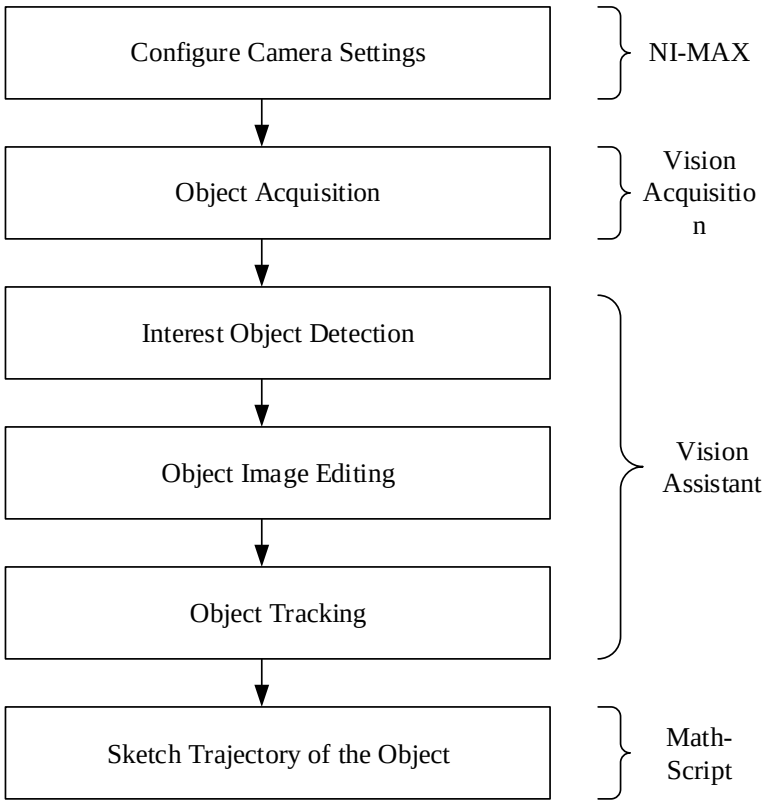


Figure 3.8: Image tracking methodology.

3.4.1 Camera Configuration

To follow the object of interest, the device (camera) should be first interfaced with a PC. The NI-MAX (Measurement and Automation Explorer) is used to configure the camera and all the settings required for object tracking. The camera attributes to be configured include: backlight compensation, brightness, contrast, exposure, gamma, hue, saturation, sharpness, and white balance; and the acquisition attributes include: mode (video), pixel format, image type, region of interest, timeout, and the package size and so forth. Here the camera object utilized for the following design is the webcam of the PC. However, for the execution reason, the camera recommended to pick is the National Instruments' smart camera.

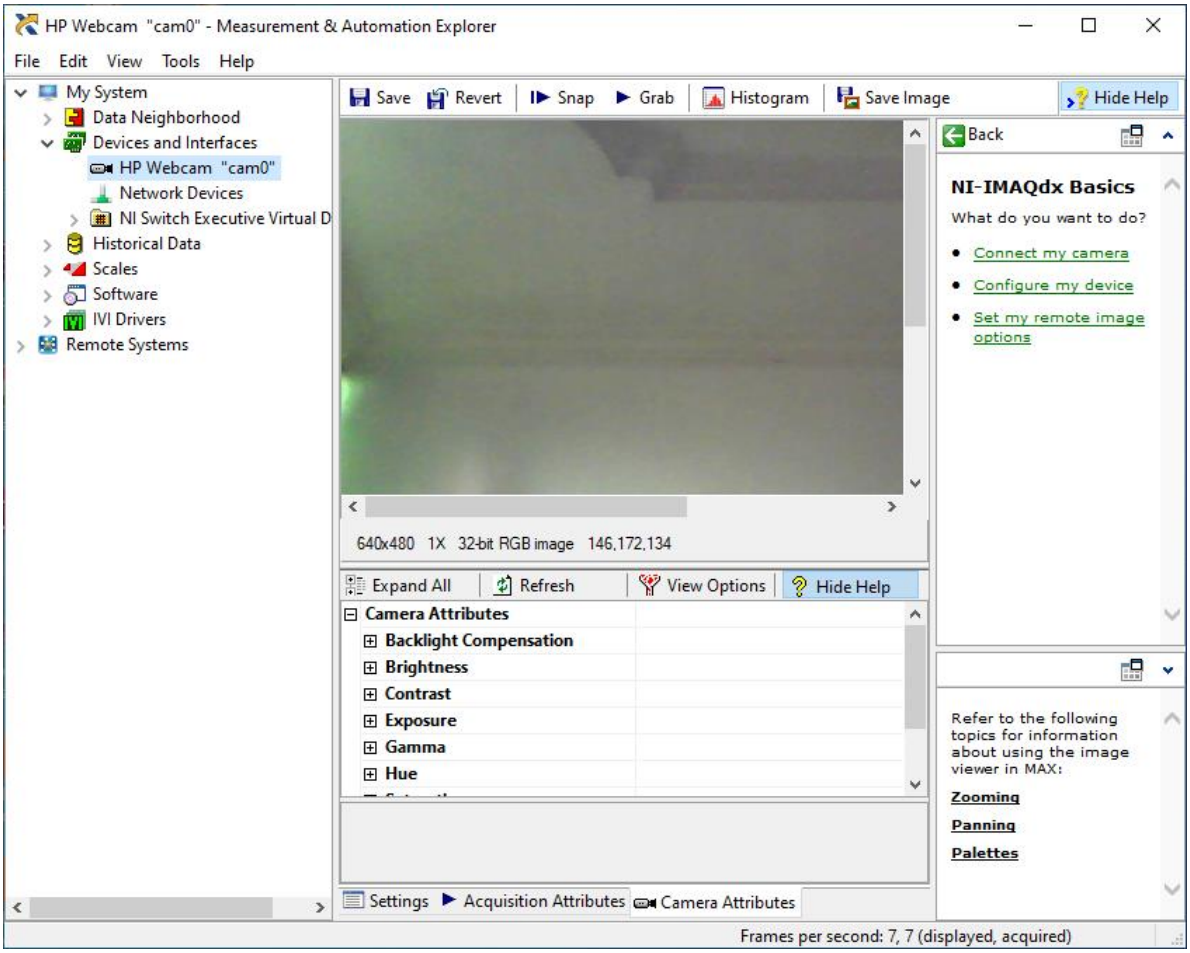


Figure 3.9: Camera configuration properties with MAX.

3.4.2 Image Acquisition

With the help of NI Vision Acquisition Express, single images or continuous images can be acquired which are going to be used for image pattern matching or image tracking purposes as required. The Vision Acquisition VI subroutine located in the block diagram is used to acquire, save and display images from the camera coupled to the computer (or the webcam of the computer) in real-time. This subroutine has some configuration settings like acquisition type (continuous acquisition with inline processing is used in this work), acquisition configuration settings (brightness, contrast, saturation, sharpness, etc.) so an adjustment has been done until a better result on the acquired image is obtained. The images acquired here can then be feed to the NI Vision Assistant for further processing.

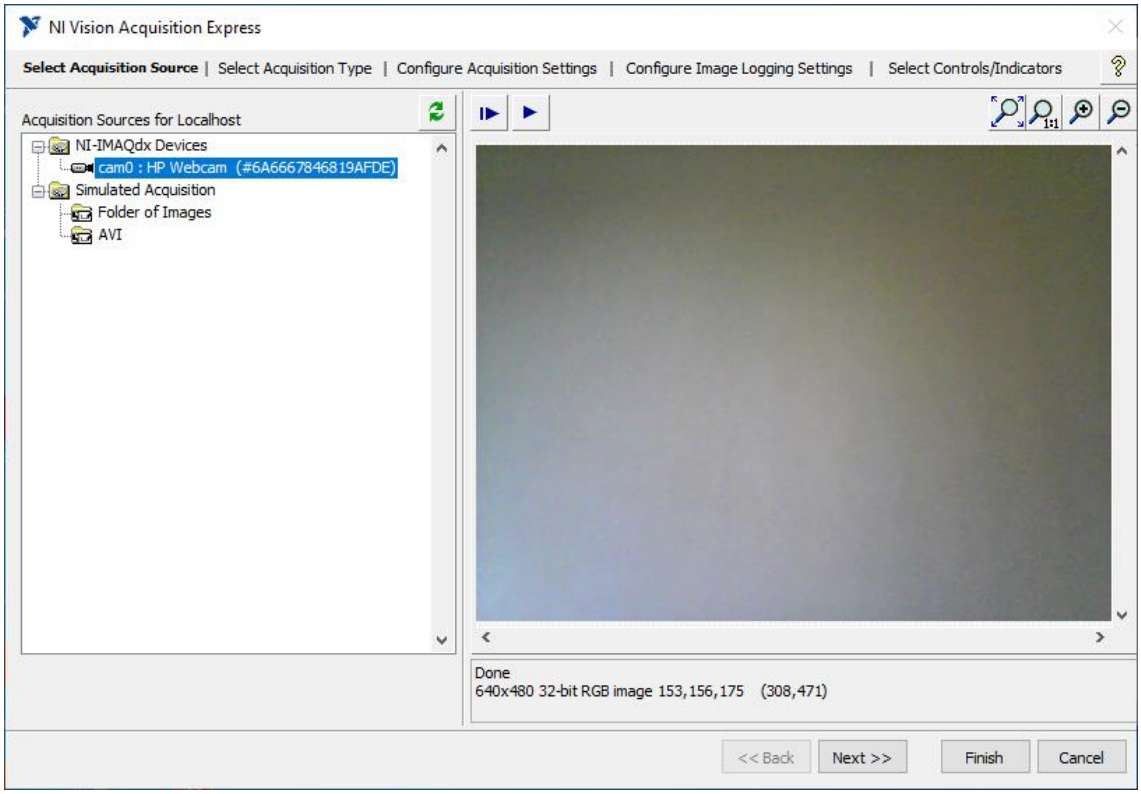


Figure 3.10: NI vision acquisition express.

3.4.3 Image Processing

The images acquired by the camera can then be processed by the Vision Assistant tool which uses a large variety of programming techniques.

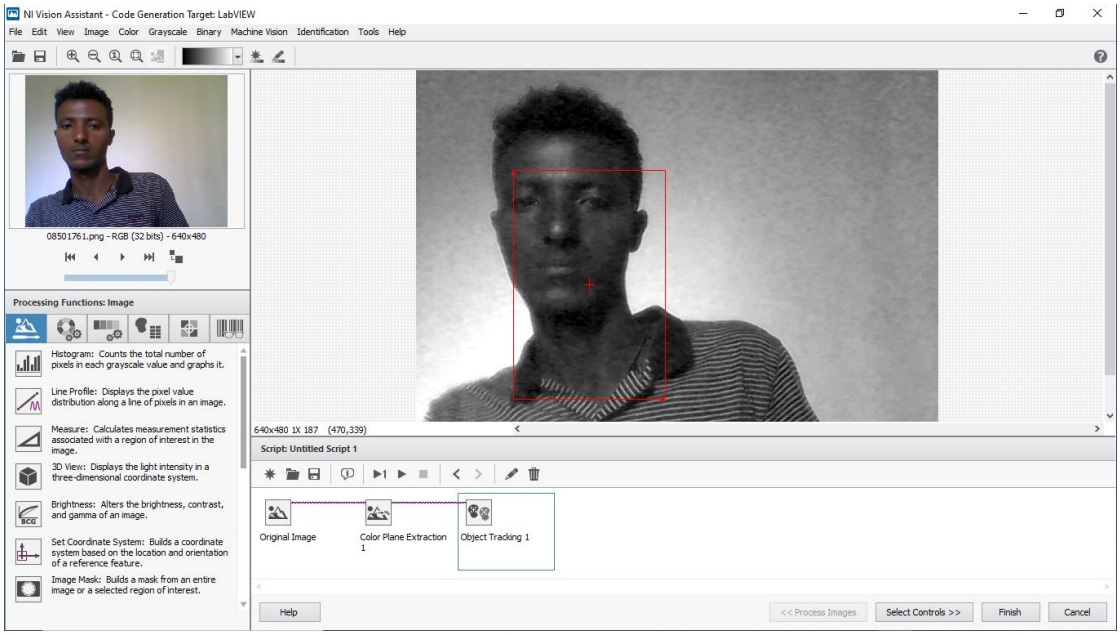


Figure 3.11: NI vision assistant.

While doing object tracking, the grayscale images are preferred over RGB color images in order to get a better efficiency as color images let the display respond slowly especially at higher resolutions [14]. The block diagram of the object tracking system is given in Figure 3.12 below.

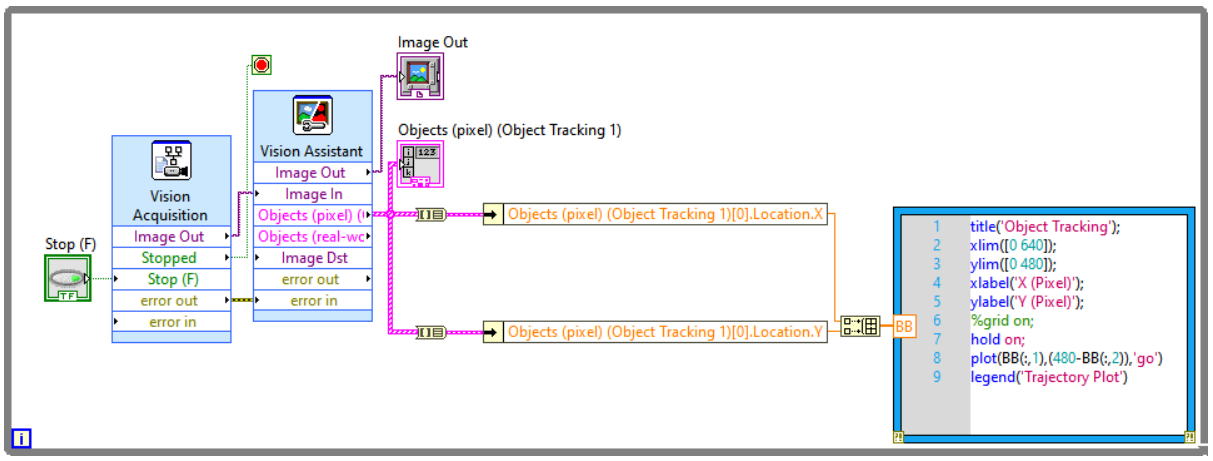


Figure 3.12: Block diagram of object tracking.

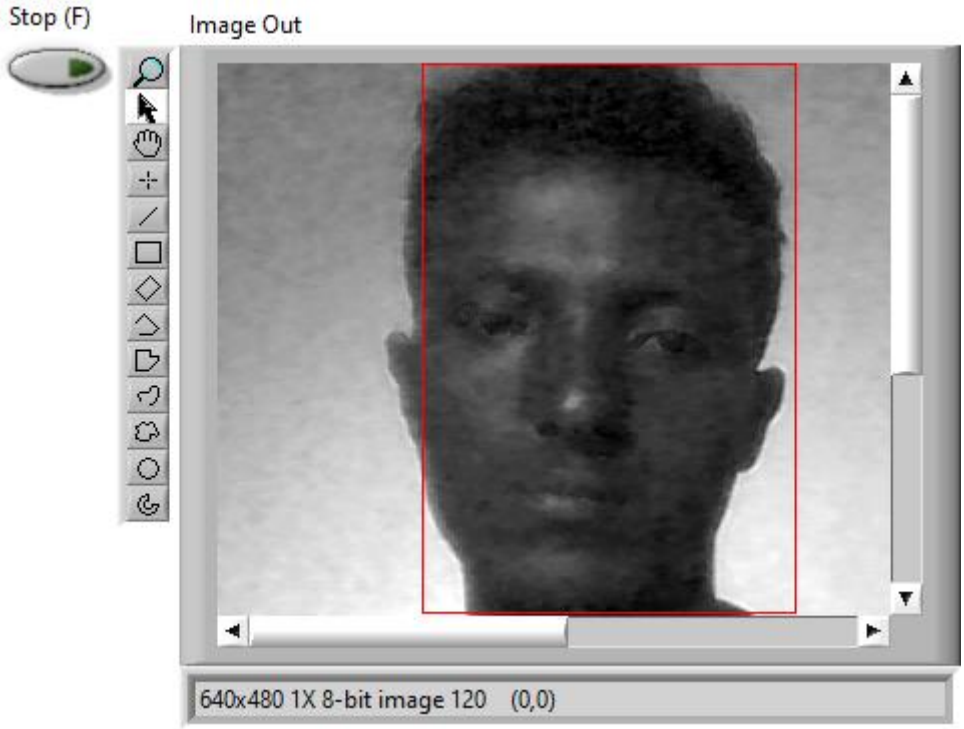


Figure 3.13: Target object chosen for tracking algorithm.

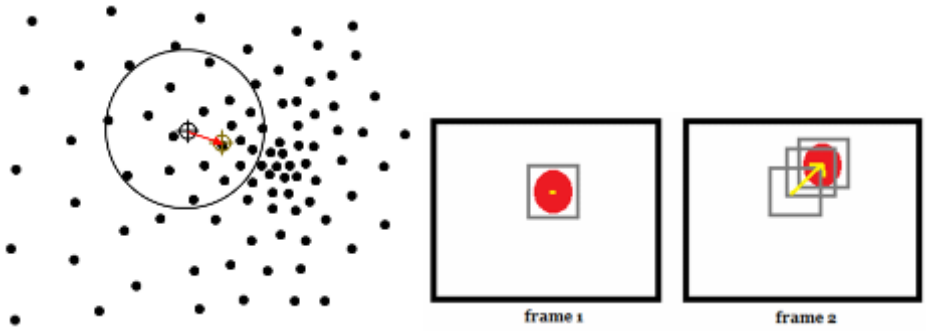
3.4.4 Tracking Algorithm

A mean-shift is a simple tracking algorithm that follows the user-defined interest objects by iteratively updating their position [45]. Conversely, shape adaptive mean shift is a broadened version of the mean-shift algorithm. In this case, not only the location but also the shape of the object including its scale is adapted frame after frame [45].

While tracking, the target object has been first described over a feature space. For representation purposes, the color histogram is used, as it is a very robust representation of the object's appearance and is chosen as a feature space. At that point, moving objects are described by their histograms. The feature-histogram-based target representations are regularized by spatial masking with an isotropic kernel [45].

The shape adaptive kernel-based mean-shift algorithm is used for the tracking purpose in order to track objects of interest. It is used to track objects that may appear to change in size or shape in a better way than the mean-shift algorithm [45].

A target object is chosen in a given frame and represented by a feature space which is a color histogram in this case with a kernel as shown in Figure 3.13 above. Then by maximizing the similarity function, a search with the current histogram and spatial data for the best target match candidate is done in the next frame [45]. The target center then moves from the current location to a new location as shown in the below figure. Until a convergence in a similarity function exists, the kernel is moved and if done, the new position of the object is updated.



(a) (b)

Figure 3.14: Mean-shift: (a) object center movement, (b) location and target update in next frame [45].

Chapter Four

Controller Design

4.1 Introduction

The principal goal of a control system is to minimize the error signals, deviations from the setpoint, as much as possible and to improve transient responses characteristics. A good controller is characterized by its stability and performance. A controller is said to be stable if in a finite time the error signals converge to a small value (or usually zero) and a good performance control system needs less time for the error signals to converge to zero.

4.2 PID Controller Design

Due to their simple structure, comprehensible control algorithm, and low cost, PID controllers are commonly utilized in industrial control applications [31]. Here our point is to design a controller whose output signals of a controlled plant can track the reference signal given as an input to the plant with the possibility of negligible error at steady-state.

$$e(t) = r(t) - y(t) \tag{4.1}$$

The schematic model of a control system with a PID controller which can incorporate the disturbance input is given below. A parallel PID controller is used in this work.

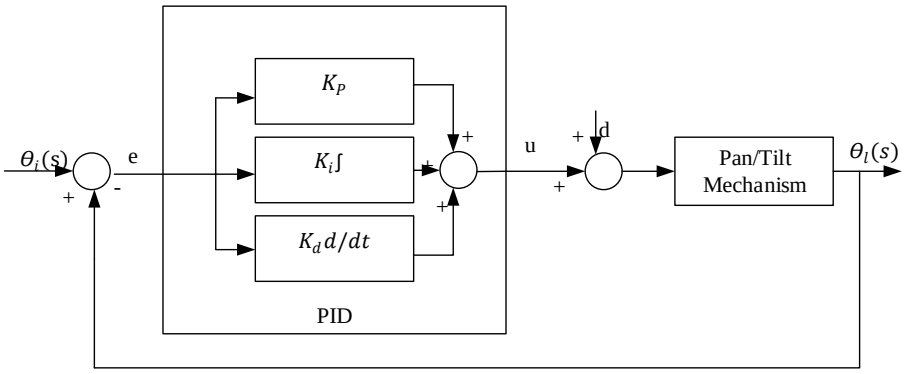


Figure 4.1: PID controller schematic model.

The Ziegler-Nicolas manual tuning rules have been used to tune the PID controller parameters. Those values obtained after employing this tuning mechanism are presented in Table 4.1 below with parameters obtained after GA-based optimization for the sake of comparison.

Table 4.1: PID controller parameters obtained after Z-N tuning and GA optimization.

Mechanism	Proportional (K_p)		Integral (K_i)		Derivative (K_d)	
	Z-N Tune	GA Tuned	Z-N Tune	GA Tuned	Z-N Tune	GA Tuned
Pan	1.6073	63.138	4.696	96.139	-0.0406	0.337
Tilt	1.351	68.86	3.899	99.847	-0.0254	0.087

The Ziegler- Nicholas tuning method used here is very time taking and does not guaranty the optimum values of the PID parameters. This has been shown by the results obtained after employing this tuning technique. In order to improve the response and optimate those parameters of a PID controller, another efficient tuning mechanism is needed. GA optimization technique is required here to see if it can handle this problem.

4.2.1 Genetic Algorithm Based PID Parameter Optimization

As discussed before, a genetic algorithm is used to generate the most effective results for optimization and search problems by employing biologically inspired operations such as crossover, selection, and mutations [43]. In addition, the GA includes some major components such as encoding schemes and fitness evaluations.

Performance Measure

The performance measure (fitness function) is selected on the basis of the controller to be tuned. The different possible performance measure criteria used to optimize a controller are:

- Transient performance-based criteria like maximum overshoot, settling time, decay ratio, etc.
- Error-based criteria such as ITAE (Integral of Time-weighted Absolute Error), IAE (Integral Absolute Error), ITSE (Integral Time Square Error), ISE (Integral Square Error), etc. Each one is a function of the error profile of the output and is given in the below equations.

$$ITAE = \int t|e|dt \tag{4.2}$$

$$IAE = \int |e|dt \tag{4.3}$$

$$ISE = \int e^2 dt \tag{4.4}$$

$$ITSE = \int te^2 dt \tag{4.5}$$

And here in this thesis, an ITAE is used as a performance measure which is used to formulate the fitness function for the GA optimization, and binary coding is applied as an encoding scheme.

Table 4.2: Selected design parameters for GA based PID optimization.

No.	Design parameter	Value
1	Population size	30
2	Number of variables	3
3	Optimization function	ITAE
4	Reproduction rate	0.2
5	Crossover rate	0.8
6	Mutation rate	0.05
7	Maximum iteration	30

The design parameters used in the simulation are shown in Table 4.2 above, while the source code developed in MATLAB is found in Appendix A of the document for the purpose of reference.

The block diagram developed for the genetic-algorithm based PID parameter optimization is given in the following figure.

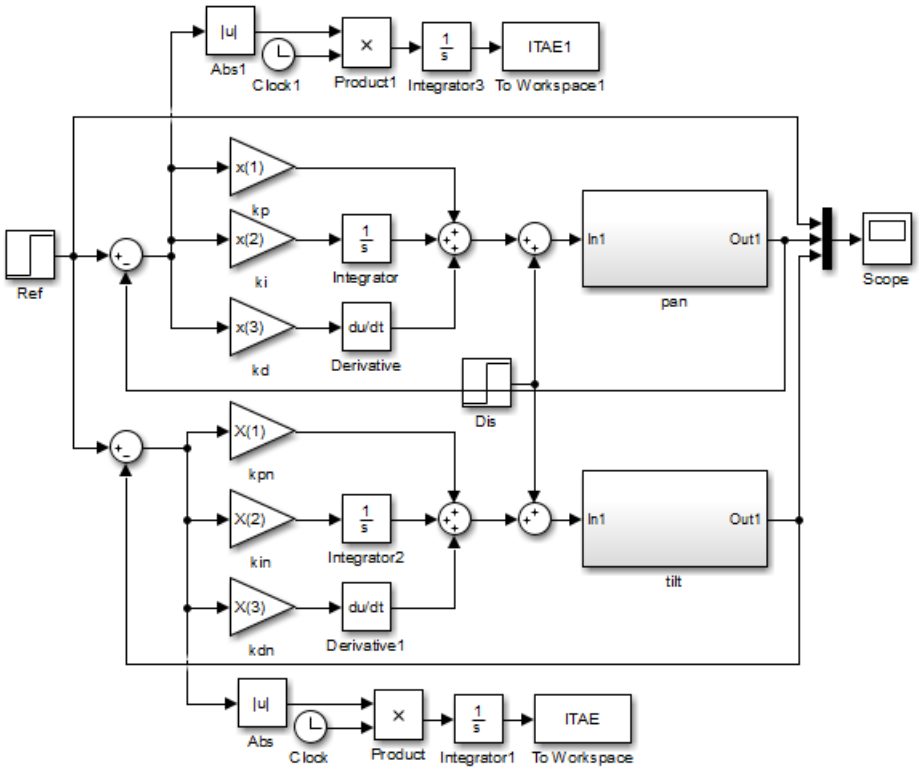


Figure 4.2: Block diagram of GA based PID parameter optimization for pan/tilt mechanism.

The step responses of the designed auto-tuning and GA-tuned PID controllers are presented in the results section of the document. The system's response without a controller, with Z-N tuned and GA tuned PID controller are given together for the comparison purpose.

Even though PID controllers can handle very well with linear control systems, they may fail to produce a robust result when they experience some non-linearity features. On the other hand, controllers like fuzzy logic produce better results when compared to conventional PID controllers. They can handle system non-linearities in a better way than PID controllers. That is why the design of the fuzzy logic controller is required here.

4.3 Design Principles of a Fuzzy Logic Controller

Here our aim is to design a Fuzzy-Logic-Controller that will drive the camera configuration output to track the constant set point so that the error goes to zero as the time approaches infinity.

$$e(t) = r(t) - y(t) \rightarrow 0 \text{ as } t \rightarrow \infty \qquad (4.6)$$

The general approaches of fuzzy logic controller design consist of three main parts and are given as follows [10].

4.3.1 The Fuzzification Module

Fuzzification transforms the physical values of the current process signal, the error signal which is input to the fuzzy logic controller, into a normalized fuzzy subset. The fuzzy subset consisting of a subset (interval) for the range of the input values and an associate membership function describing the degrees of the confidence of the input belonging to this range [10]. In this module we are going to create, linguistic variables that represent in words the input and output variables of our system and membership functions which are numerical functions that correspond to linguistic terms [10].

Creating Linguistic Variables

While creating the linguistic variables to represent an input or output variable, identifying the number of linguistic terms and categories of the values of the linguistic variable is necessary [10, 12]. Usually, the number of linguistic terms of a linguistic variable is set to be odd and are symmetrical about the center linguistic term at each extreme. Three to seven linguistic terms are adequate in most applications, for categorizing the values of a linguistic variable [12].

Our goal here is to design Fuzzy Logic Control in order to minimize the error between the camera center and the object center. And hence, we use the error e(t) and change of error $\dot{e}(t)$ as the input variables to the controller and u(t) as the output variable. The following two equations give the required system equations.

$$e(t) = r(t) - y(t) \tag{4.7}$$

$$\dot{e}(t) = (e(t) - e(t-1))/\Delta t \tag{4.8}$$

Where r(t) is desired camera center and y(t) is object center.

The number of linguistic terms for both the input and output variables here is taken to be five, i.e., the deviation of the center point of the camera to the object center can be adjusted by giving the appropriate signals to the pan and tilt servo motors. The linguistic terms for camera position in both x and y direction, i.e., for pan motor and tilt motor, can be defined as Left (L), Left Center (LC), Center (C), Right Center (RC), Right (R), and Down (D), Down Center (DC), Center (C), Up Center (UC) and Up (U) respectively. The corresponding output variable linguistic terms can be defined as NL (Negative Large), NS (Negative Small), Z (Zero), PS

(Positive Small), and PL (Positive Large). Error Negative Large (ENA), Error Negative Small (ENB), Zero (Z), Error Positive Small (EPB), and Error Positive Large (EPA) linguistic terms are used for the change of error input variable. The region of interest of the camera mapped to motor orientations is given in Figure 4.3 below. The pan motor mapped to the horizontal movement direction while the tilt motor corresponds to the vertical movement of the interest object.

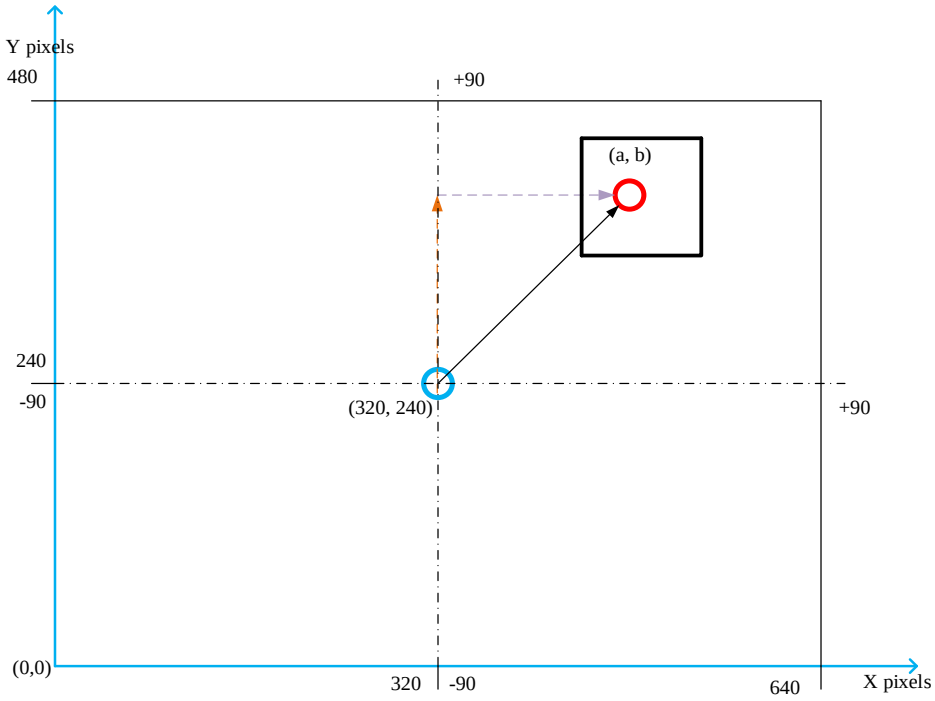
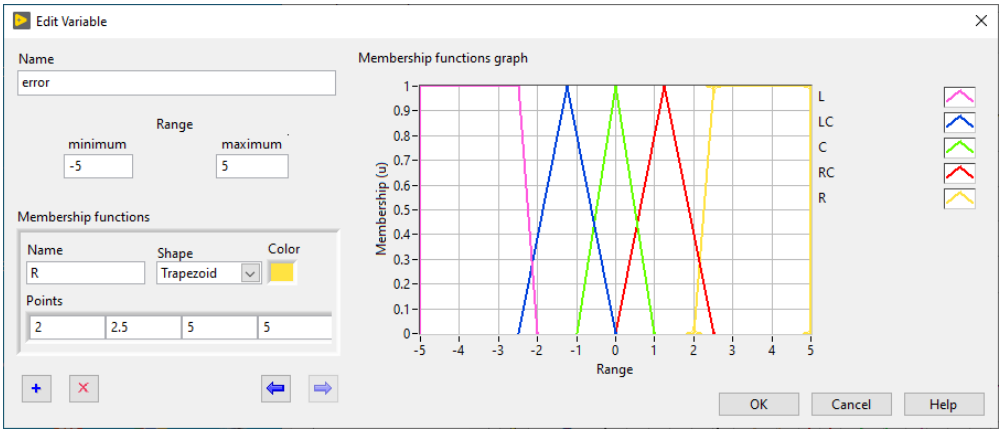


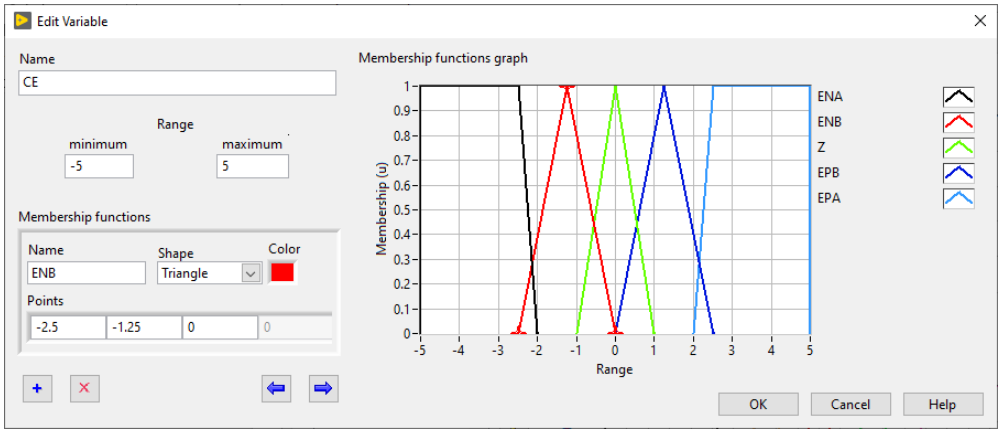
Figure 4.3: Region of interest mapped to motor orientations.

Creating Membership Functions

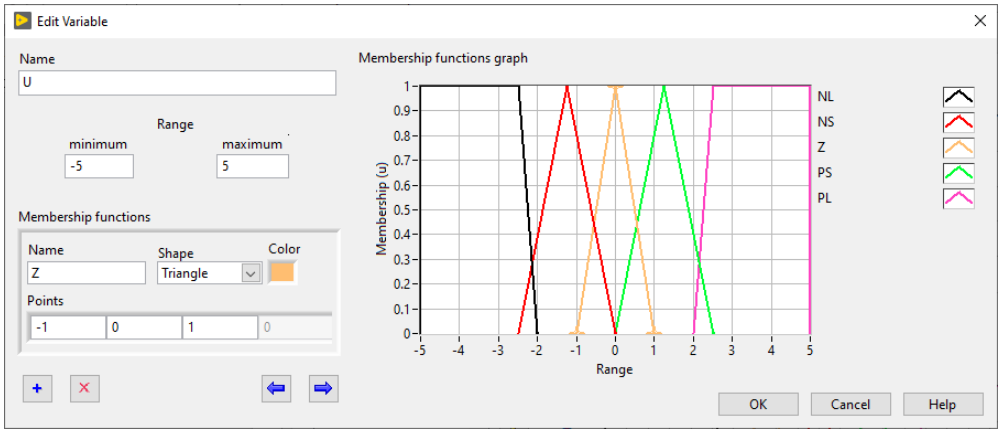
The numerical functions corresponding to each linguistic term have been created by applying the normalized standard membership functions (Triangular Type and Trapezoidal Type) and illustrated in the following figures.



(a)



(b)



(c)

Figure 4.4: Membership functions of pan motor; (a) error, (b) change of error and (c) output.

Note that, the membership functions of the tilt mechanism are created in a similar fashion as a tilt motor.

4.3.2 The Fuzzy Logic Rule Base

The rule base of a FLC should be designed with more attention since it determines the satisfactoriness of the given controller for the application needed. A general procedure used while designing a fuzzy logic rule base includes the following [10]:

1) Identifying the Process States and Control Variables.

   The set-point r here is the desired object position in pixel, r = 320 for x motor and r = 240 for y motor, and the overall controlled system output is y(t), objects current position in pixels. The control variable u is created from the error signal through the controller.

2) Identifying Input Variables to the Controller.

   The error signal e(t) is one of the input variables to the controller. And in order to build an efficient and complete rule base, more auxiliary input variables are needed. Hence, to write an IF-THEN control rule the error signal e(t) is not sufficient. Therefore, in this particular work the change of error is used as a second input variable to the controller; whereas more input variables such as the second derivative and/or the sum of the errors can be used as an input variable to make the controller more effective for other applications.

   From the principles of Calculus that if a curve is moving up, the derivative of the curve is positive and if it is moving down its derivative is negative. Hence, the change of the error signal, denoted $\dot{e}$ & or Δe, can help us to distinguish the two situations at points a and d, as well as at points b and c, of Figure 4.5.

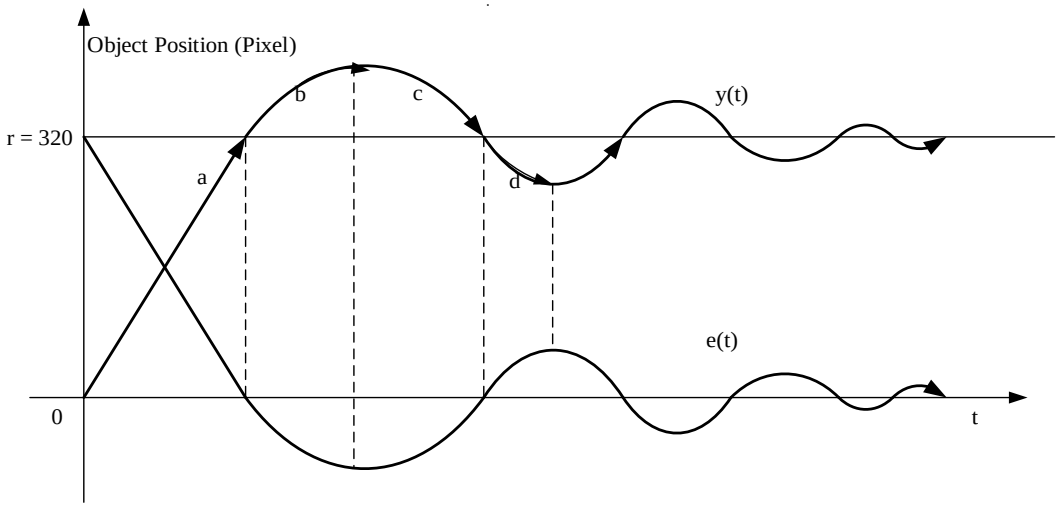


Figure 4.5: Object position set-point tracking.

3) Creating a Fuzzy Logic IF-THEN Rule Base

While creating the fuzzy logic IF-THEN rule base, the four situations (a, b, c and d) shown in Figure 4.5 above have been considered. The control rules have been formulated as follows:

R^1: IF e > 0 AND $\dot{e}$ < 0 THEN u(t+) = u(t), (position a on the Figure)

R^2: IF e <0 AND $\dot{e}$ < 0 THEN u(t+) = -u(t), (position b on the Figure)

R^3: IF e < 0 AND $\dot{e}$ > 0 THEN u(t+) = u(t), (position c on the Figure)

R^4: IF e > 0 AND $\dot{e}$ > 0 THEN u(t+) = -u(t), (position d on the Figure)

Where u(t+) = u(t) tells the controller to retain its previous action and u(t+) = -u(t) tells the controller to change its previous action to the opposite. To have an efficient controller the above four control rules are not sufficient and additional control rules have been added. If the output is far away from the reference signal, then the control action will be large and if the output is close to the reference, then the control action is small. And doing this, the possible total number of rules for a given fuzzy system [10, 11, 12] is given by the following equation:

$$N = P_1 \text{ x } P_2 \text{ x } \dots \text{ x } P_n \quad (4.9)$$

Where P_n is the number of linguistic terms for the input linguistic variable n. Using this, we are going to have 5 x 5 = 25 total number of rules and are given in Table 4.3 below.

Table 4.3: A complete rule base used.

AND		Change of error				
		ENA	ENB	Z	EPB	EPA
Error	L	NL	NL	NS	NS	PL
	LC	NS	NS	NS	NS	PS
	C	Z	NS	Z	Z	Z
	RC	PS	Z	PS	NS	NS
	R	PL	PS	PS	NS	NL

4) Specifying an Antecedent Connective [12]

For rules with more than one antecedent, an antecedent connective should be determined in order to calculate the truth value of the aggregated rule antecedent. The most common antecedent connectives are given below.

AND (Minimum): $\mu(A \bullet B) = \min(\mu A, \mu B)$, intends to utilize the minimum degree of membership of the antecedents as the truth value of the aggregated rule antecedent.

AND (Product): $\mu(A \bullet B) = (\mu A \times \mu B)$ intends to utilize the product of the degrees of membership of the antecedents.

OR (Maximum): $\mu(A+B) = \max(\mu A, \mu B)$ specifies to use the largest degree of membership of the antecedents.

OR (Probabilistic): $\mu(A+B) = ((A+B) - (AB))$ specifies to use the probabilistic sum of the degrees of membership of the antecedents.

Notice that these definitions agree with the logical operators used in Boolean logic. A truth table uses conventional operators to yield equivalent results. Considering all the above inference methods, the AND (Minimum) antecedent connective is used in this work which gives us a better result.

4.3.3 The Defuzzification Module

After creating a rule base of a fuzzy control system, it is a must to specify how a fuzzy controller performs Defuzzification for the system. It is the reverse of the fuzzification module. Defuzzification converts all the fuzzy terms created by the rule base of the controller to crisp terms or numerical values. After then it sends them to the physical system which is a plant or process in order to execute the control of the system. While selecting a defuzzification method, it is necessary to consider the context of the decision we want to calculate with the fuzzy logic controller. For quantitative decisions like project prioritization, the CoM method is better than others and for qualitative decisions, such as an evaluation of creditworthiness MoM is the efficient method [12].

In this work, a Center of Area (CoA) defuzzification method is used, as it is usually applied for closed-loop control applications and generates better results. The designed Fuzzy Logic Controller for pan/tilt mechanism is shown in Figure 4.6 below. The corresponding results of the designed controller are given in the simulation and results section of the document.

The simulation results showed that the fuzzy-logic controller is still inefficient to generate efficient results. Especially it is very difficult to obtain the appropriate operating ranges and nonlinear factors of each variable. In the next section, we are going to design the Fuzzy-PID controller where the operating ranges of the variables are obtained by genetic algorithm-based optimization. The inputs of the fuzzy-logic controllers are set to be the proportional, integral,

and derivative parts of the conventional PID controller. The genetic algorithm here is utilized to optimize the operating ranges of both the inputs and output of a fuzzy logic controller.

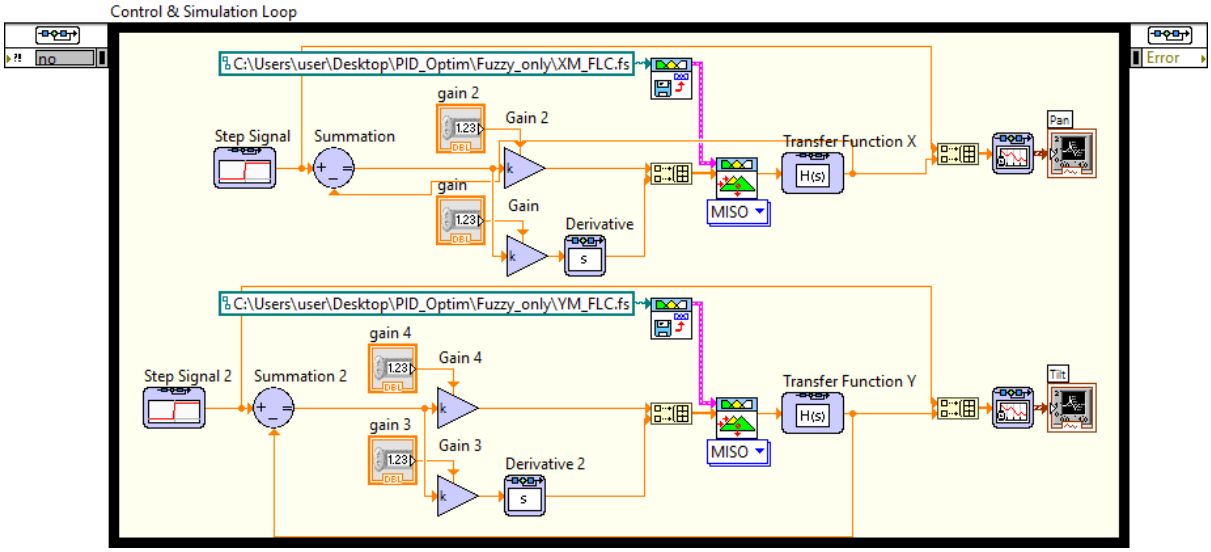


Figure 4.6: The proposed fuzzy logic control system for the pan/ tilt mechanism.

4.4 Optimal Fuzzy-PID Controller Design

Fuzzy Logic Controller design mainly focuses on the type of FLC, the number and shape of *MFs*, and the fuzzy rules. Since the design techniques of conventional linear PID controllers have matured, using GA to optimize the Fuzzy-PID controller design will give a better result and is advantageous [29]. The GA is used here in order to find the optimum parameters of a system.

In Fuzzy-PID Controller Design, the controller structure should be considered in advance. The fuzzy control rules should follow conventional PID controller and in order to improve system performance, membership function tuning is performed [29]. Here the shape of the *MFs* is set to be a symmetrical shape. There are also other methods like readjustment of symmetrical membership functions to asymmetrical membership functions which still produce improved system performance [29]. In this thesis, the operating ranges are tuned, which are useful parameters that describe the equivalent FLC design from a PID controller.

Fuzzy Variables

The three inputs e(t), $\int e(t)$, $\dot{e}$(t) and the output u(t) of the conventional PID controller are used as fuzzy variables in the fuzzy logic design as given in Figure 4.7 below. The operating ranges

for each of the variables are assumed to be: e(t) = [-a_e, a_e], $\int e(t)$ = [-a_i, a_i], $\dot{e}$(t) = [-a_d, a_d], u(t) = [-a_u, a_u].

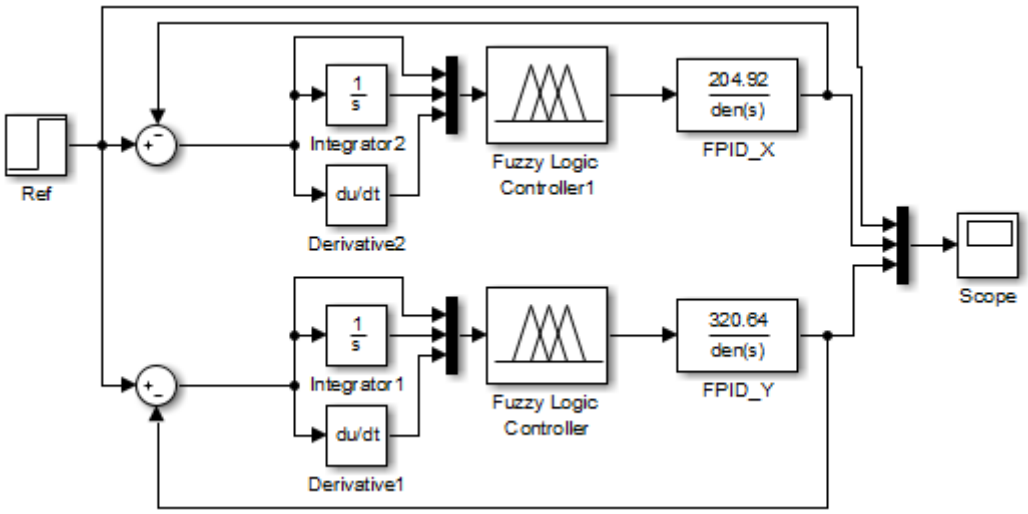


Figure 4.7: FPID controller for pan/tilt mechanism.

For input fuzzy variables, each variable is represented by five triangular-shaped and equally spaced membership functions, and the output variable is represented by thirteen singleton membership functions. Note that, the fuzzy type used here is a Sugeno-type over Mamdani-type. The main difference between the two is that in Sugeno, output *MFs* are either linear or constant. The digital hardware design of Mamdani Fuzzy Controllers is long and complex [48].

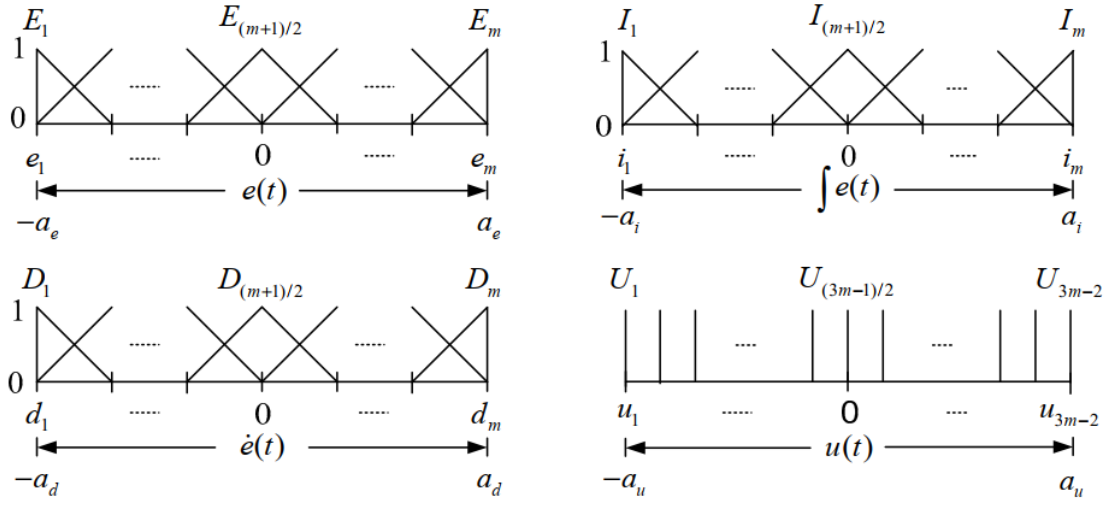


Figure 4.8: Graphical definition of membership functions for fuzzy variables [29].

The overall fuzzy rules are represented by the sliced cube fuzzy associative memory (FAM) [29] as shown in Figure 4.9 below. Each fuzzy rule is defined as:

If e(t) is E_i and $\int e(t)$ is I_j and $\dot{e}$(t) is D_k THEN u(t) is U_l, l = i+j+k-2 (4.10)

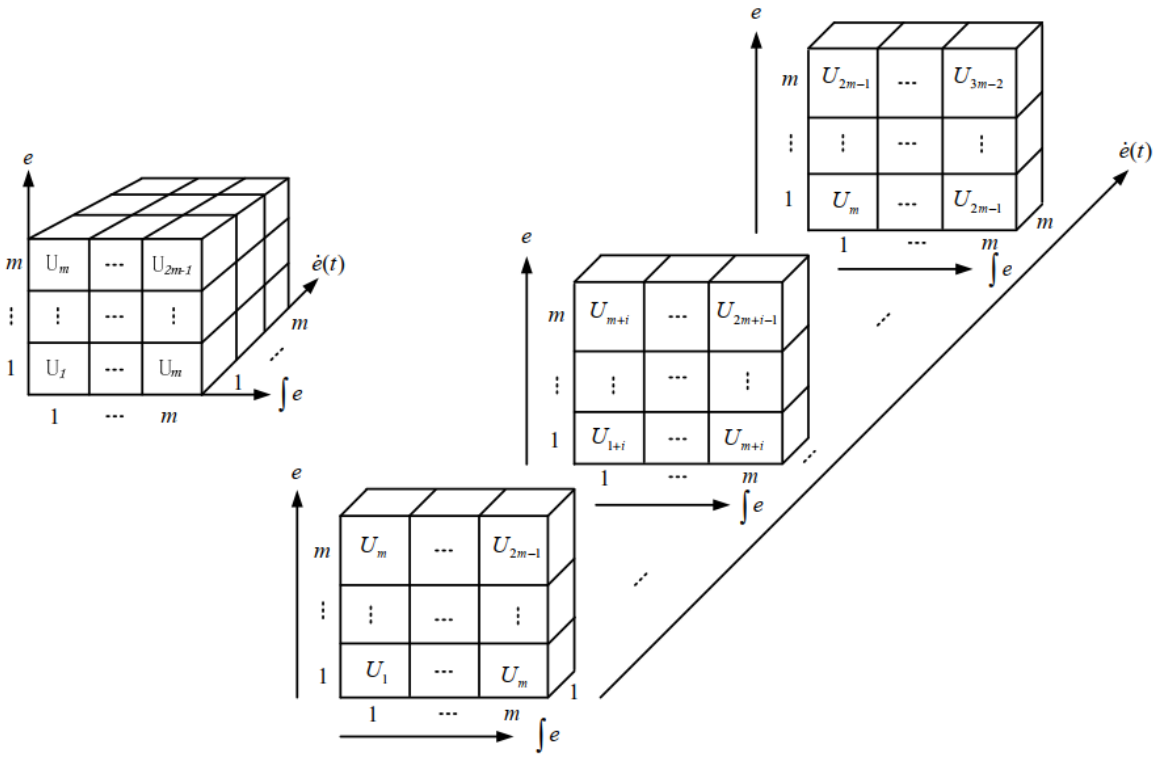


Figure 4.9: Sliced cube FAM representation of the knowledge base [29].

And the crisp output u(t) of the corresponding equivalent fuzzy logic controller is given by [29]:

$$u(t) = \frac{a_u}{3a_e}e(t) + \frac{a_u}{3a_i}\int e(t)dt + \frac{a_u}{3a_d}\dot{e}(t) \quad (4.11)$$

Which gives us a linear PID controller with:

$$K_P = \frac{a_u}{3a_e}, K_I = \frac{a_u}{3a_i}, and\ K_D = \frac{a_u}{3a_d} \quad (4.12)$$

Equation (4.12) shows us it has no relation with m (number of membership functions). Here in the Fuzzy-PID controller which is the equivalent of the conventional PID controller, we have four parameters for tuning. Those parameters to be optimized or tuned are: a_e, a_i, a_d and a_u. While designing an optimal Fuzzy-PID controller, one can try to optimize the scaling factor, the nonlinear factor, and/or the rule base depending on the application used and design parameter requirements. Even though optimization of the nonlinear factor and rule bases

generates improved results, the computation time becomes too long and the complexity of the system increases too [29]. The fuzzy inference system, rule base view, and surface view of the FPID controller are given in the below figures.

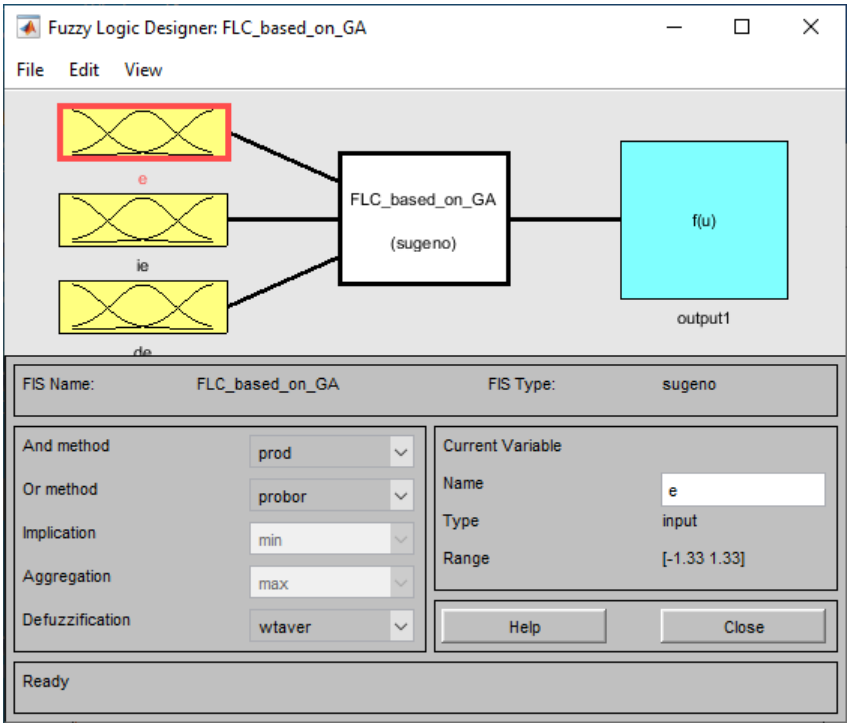


Figure 4.10: Fuzzy inference system.

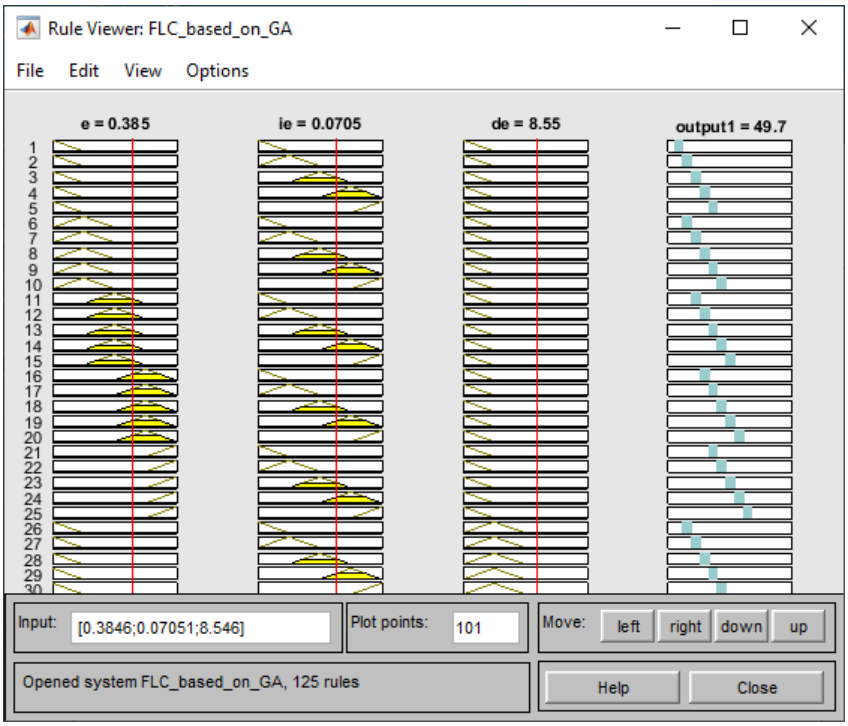


Figure 4.11: Rule base view.

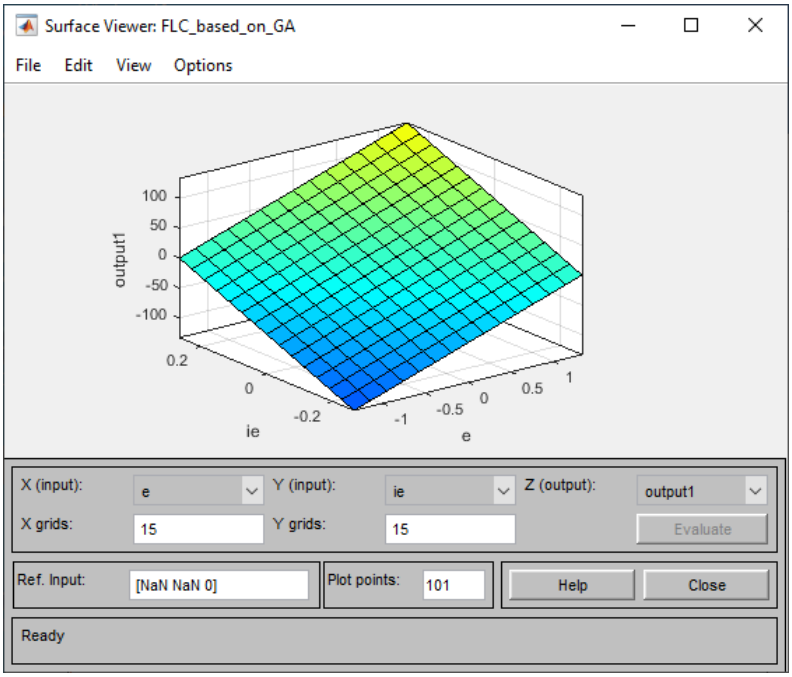


Figure 4.12: Surface view.

The parameters used for genetic algorithm-based Fuzzy-PID parameter optimization are tabulated in Table 4.4 below, while the code developed for the optimization purpose is presented in the Appendix section of the document.

Table 4.4: Selected design parameters for GA based FPID optimization.

No.	Design parameter	Value
1	Population size	30
2	Number of variables	5
3	Membership function type	Triangular, singleton
4	No. of input fuzzy sets	5
5	No. of output fuzzy sets	13
6	No. of input/output variables	3/1
7	Optimization function	ITAE
8	Reproduction rate	0.2
9	Crossover rate	0.8
10	Mutation rate	0.05

The proposed genetic algorithm tuned Fuzzy-PID controller block diagram is given in the following figure. The operating range scaling factors for the inputs and output are inside the fuzzy logic source code and can be referenced in the Appendix section of the document.

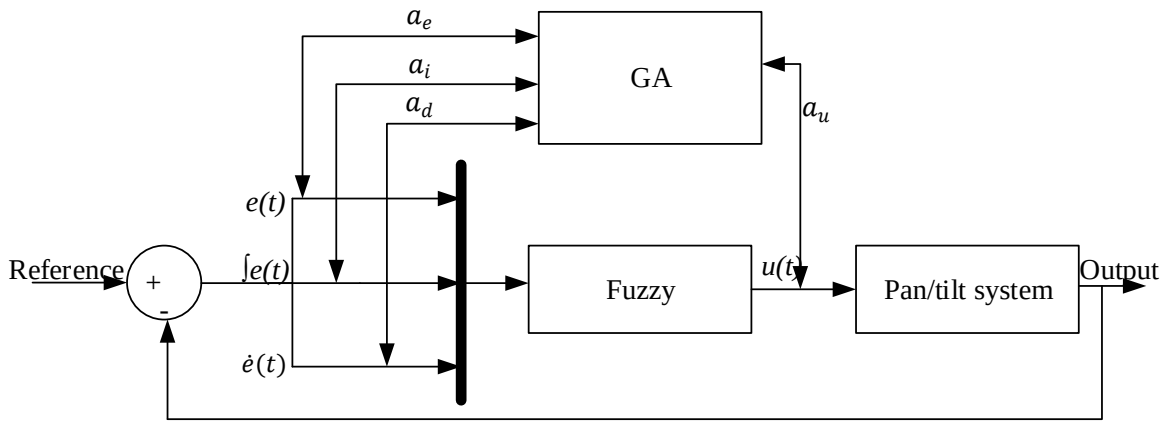


Figure 4.13: The proposed optimal FPID controller.

The corresponding simulation results of the designed FPID controller are illustrated in the results and discussion section of the document. There, the simulation results of the conventional PID, Fuzzy Logic, and Fuzzy-PID controllers have been discussed.

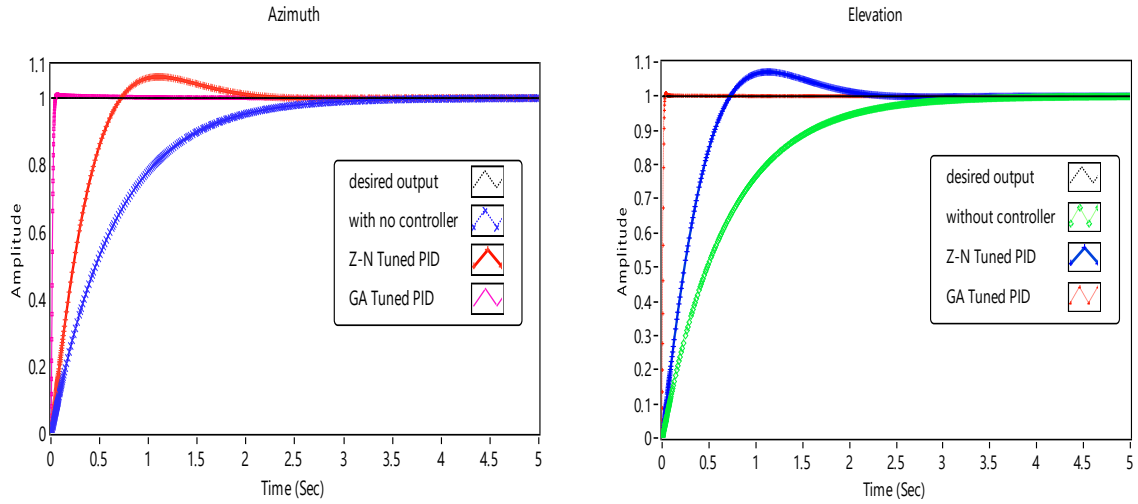
Chapter Five

Simulation Results and Discussion

Object tracking is a broad concept consisting of many system components. This thesis does not attempt to implement a full real-time object tracking system. The goal of this thesis was to develop an object tracker in a two-dimensional (2-D) plane both in static and dynamic cameras. The vertical and horizontal movements of the tracker are designed to be at their possible efficient values. This chapter consists of the simulation results and discussion parts of the thesis. In the first section, the performance evaluation of designed controllers is presented. Next is the static camera-based object tracking that has been given. Finally, dynamic camera-based object tracking is presented.

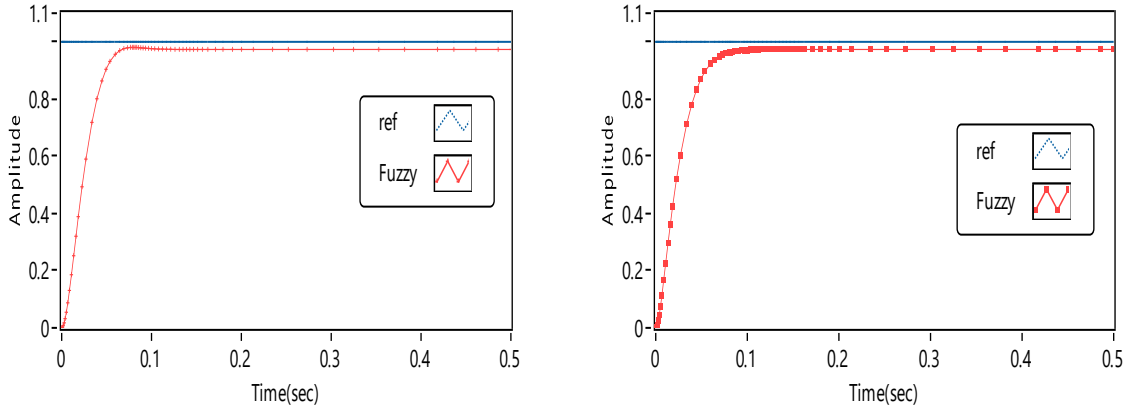
5.1 Performance Evaluation of Designed Controllers

The corresponding step responses of the designed controllers are given as follows. In Figures 5.1 and 5.2 the step responses of the no-controller profile, Z-N Tuned PID, GA tuned PID and fuzzy logic controller response plots are given.



(a) (b)

Figure 5.1: PID unit step response of: (a) the azimuth system and (b) the elevation system.



(a) (b)

Figure 5.2: Fuzzy unit step response of: (a) the azimuth and (b) elevation system.

As Figures 5.1 and 5.2 above showed the responses of the fuzzy logic controller given are better than the corresponding conventional PID controller. The fuzzy logic controller response has least overshot, least settling time, and minimum rise time but has a steady-state error when compared to the conventional PID controller. As PID controllers are poor at handling non-linear systems, the fuzzy logic controllers are capable of generating good results with respect to conventional PID controllers. But fuzzy logic controllers are not still efficient and another method of controller design technique, FPID, is proposed. When compared to the fuzzy logic controller designed response, the GA tuned PID controller generated a very good result. It has fast rising time, minimum overshoot, and minimum settling time. Therefore, GA tuned PID response is better than the fuzzy logic controller designed in this case. Again, a better result has been obtained after implementing a fuzzy logic controller with PID and optimizing it with a genetic algorithm.

Figures 5.3 and 5.4 showed the response plots of the FPID controller and GA tuned FPID controller. The FPID controller response had an overshot, big rising time, and settling time when compared to the corresponding GA-tuned FPID controller. The proposed GA-tuned FPID controller shown in Figure 5.4 has given improved results when compared to the previous controllers. The response has minimum overshoot, negligible steady-state error, and a very short rising and settling time as shown in Tables 5.2 and 5.3 below. Therefore, the GA-tuned FPID controller has given a speedy response with a minimum steady-state errors.

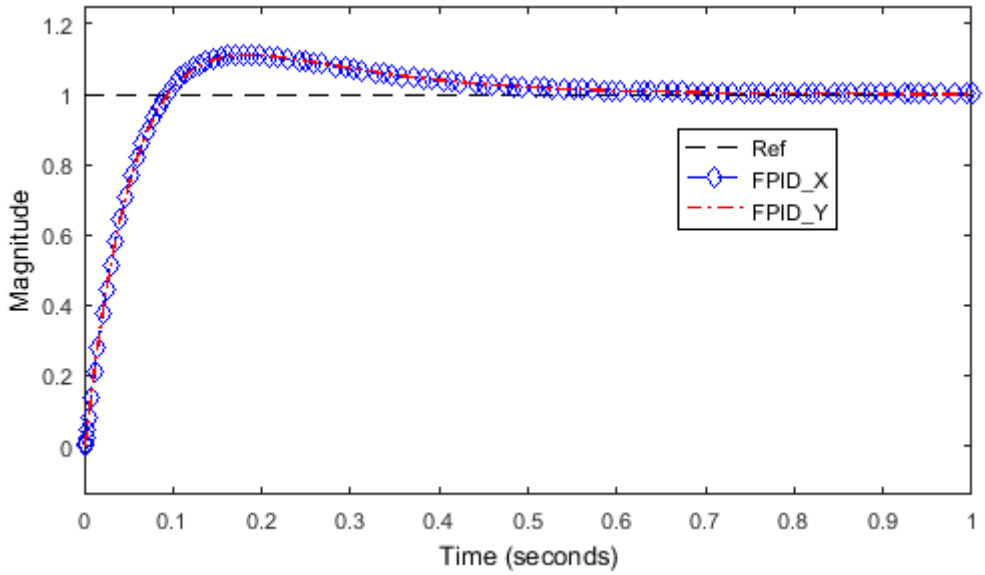
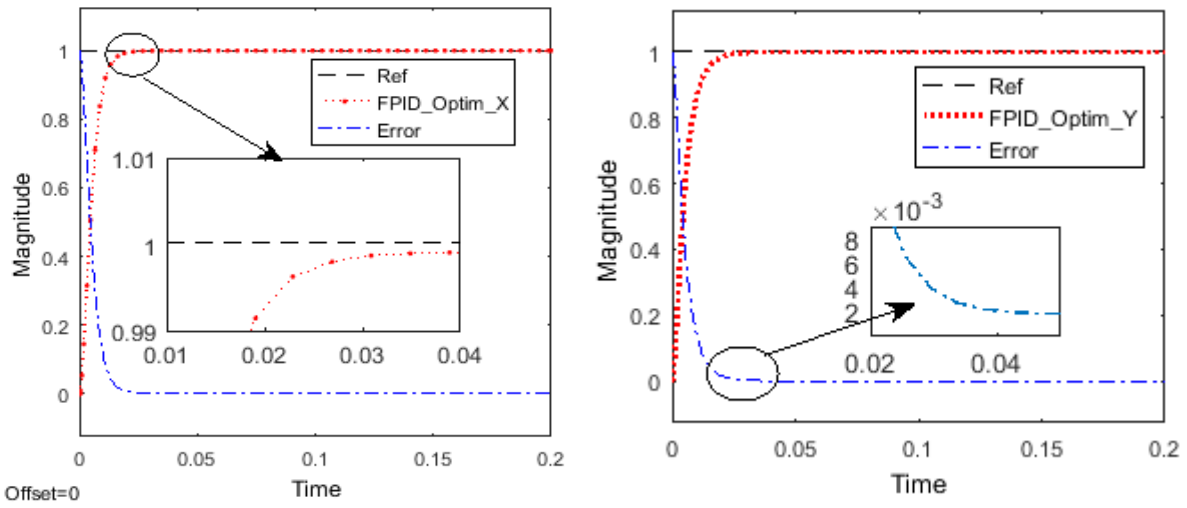


Figure 5.3: Unit step response of FPID controller before GA optimization.

The parameters obtained after GA optimization for each case are tabulated in Table 5.1 below.

Table 5.1: Tuned FPID parameters.

Parameter	K_1	K_2	K_3	K_4	K_5
Azimuth	3.01	108.91	759.64	4752.19	3169.12
Elevation	3.002	1157.456	948.234	4582.13	120.224



(a) (b)

Figure 5.4: Unit step response of the GA tuned FPID system for (a) azimuth and (b) elevation.

In Figures 5.5 and 5.6 the GA-tuned PID and GA-tuned FPID controller step responses for the pan/tilt mechanism are plotted. As shown in Tables 5.2 and 5.3, the GA-tuned FPID controller had given comparatively better results over the GA-tuned PID controller. The rise time reduced

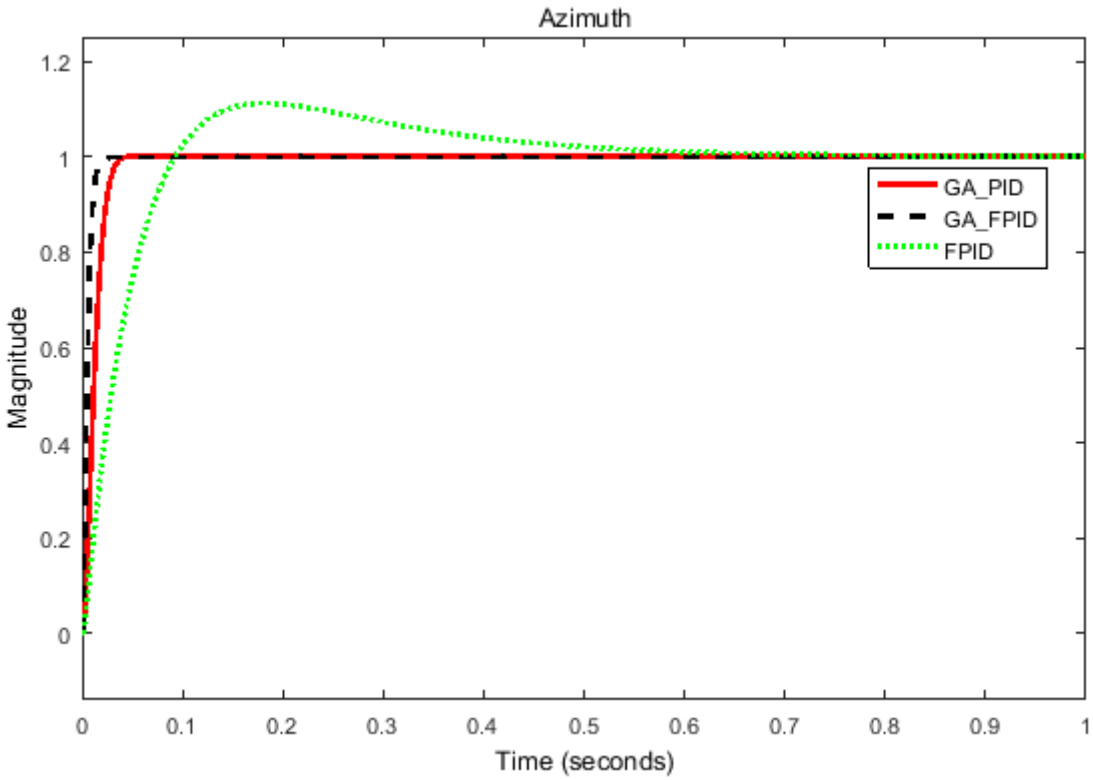


Figure 5.5: GA tuned PID and GA tuned FPID responses of azimuth system.

from 20.2 to 9.4 and 16.9 to 7.9 milliseconds (ms) for the pan/tilt system respectively. The settling time is dropped from 34 to 16 and 27 to 15 milliseconds (ms) for the pan/tilt system respectively. The steady-state error is negligible for each case and the maximum overshoot has been reduced from 11.2 to 0.1 and 0 for GA PID and GA FPID controllers respectively for pan motor.

Table 5.2: Transient and steady-state response parameters for azimuth.

Controller	Transient and steady-state response parameter (Pan)			
	T_r (ms)10-90%	T_s (ms) $\pm 2\%$	Steady-state error	Overshoot (%)
GA_FPID	9.4	16	$3.973*10^{-5}$	0
GA_PID	20.2	34	$6.775*10^{-5}$	0.1
FPID	65.6	500	$7.130*10^{-4}$	11.2

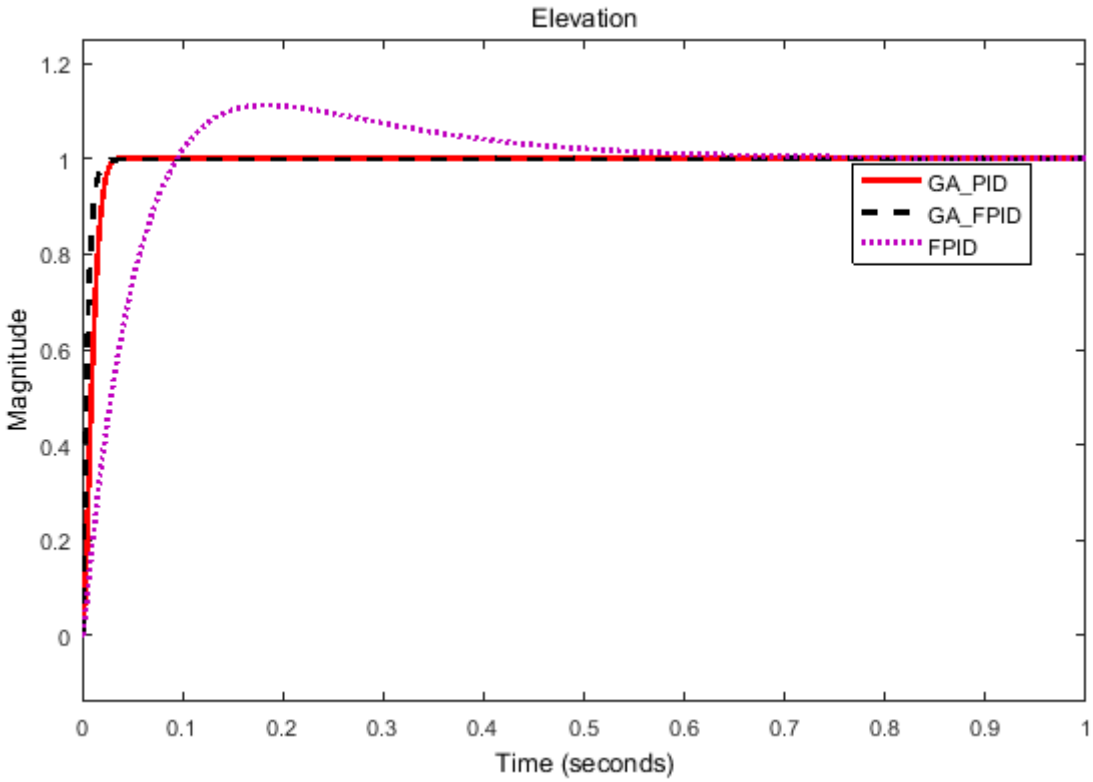


Figure 5.6: GA tuned PID and GA tuned FPID response of elevation system.

Table 5.3: Transient and steady-state response parameters for elevation.

Controller	Transient response parameter (Tilt)			
	T_r (ms)10-90%	T_s (ms) $\pm 2\%$	Steady state error	Overshoot (%)
GA_FPID	7.9	15	$1.466*10^{-5}$	0.1
GA_PID	16.9	27	$1.363*10^{-4}$	0.2
FPID	66.75	512	$1.016*10^{-3}$	11.0

Based on these results, therefore, the genetic algorithm tuned Fuzzy-PID controller achieved better results than the genetic algorithm tuned PID controller. It is then used for the tracking mechanism in order to effectively control the position of the pan/tilt system.

The corresponding actuating signals for the designed controllers are given below. As the below figures show, the control voltage for the controller GA tuned FPID is big which may go beyond the ratings of our actuators and may tend to damage them.

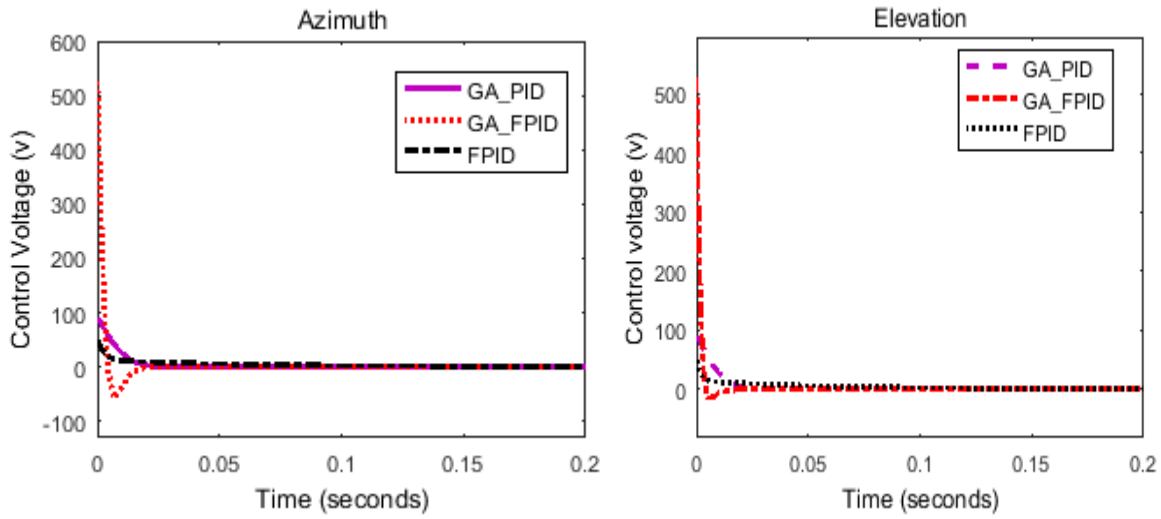


Figure 5.7: The actuating signals due to a step input.

Hence, saturation is used so as the control voltage is not increased beyond the actuator's ratings or natural capabilities. The response of the actuating signals due to a step input in the case of GA tuned FPID controller with saturation incorporated are given below.

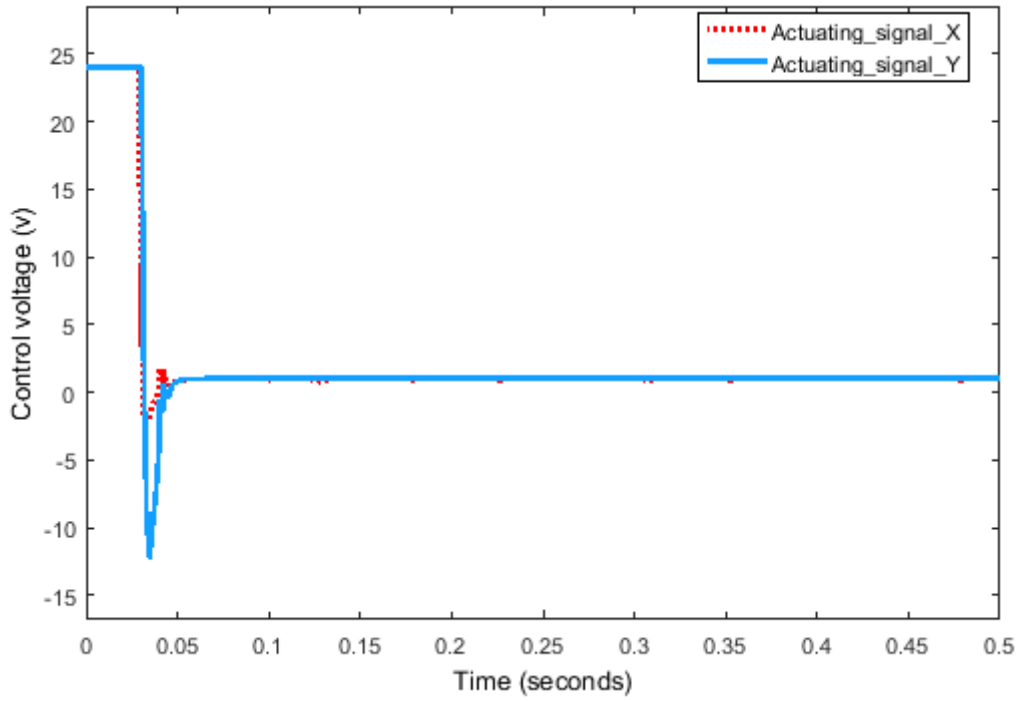


Figure 5.8: The actuating signals due to a step input with saturation.

The response of the system with saturation tracks the set point with $3.095*10^{-3}$ and $9.487*10^{-4}$ steady-state error for x and y respectively and given in Figure 5.9 below.

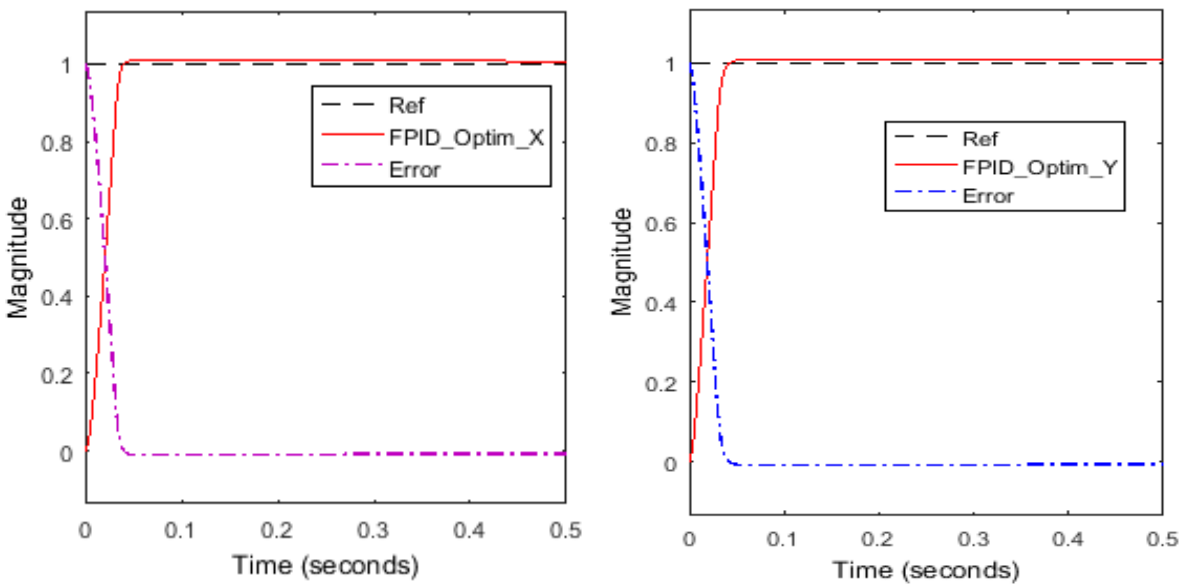


Figure 5.9: GA tuned FPID controller response with saturation.

External Disturbance Rejection Capability

The performance of the controllers with external disturbance are tested by additively applying a pulse signal of amplitude 2, frequency 1 rad/sec and pulse width of 10 % at the input to the plant. Due to symmetry only the controllers of the azimuth motor are discussed here. The performance of the design controller with an external disturbances like wind and dust is shown in the below figure. As the simulation results showed the GA tuned Fuzzy-PID controller has a better performance.

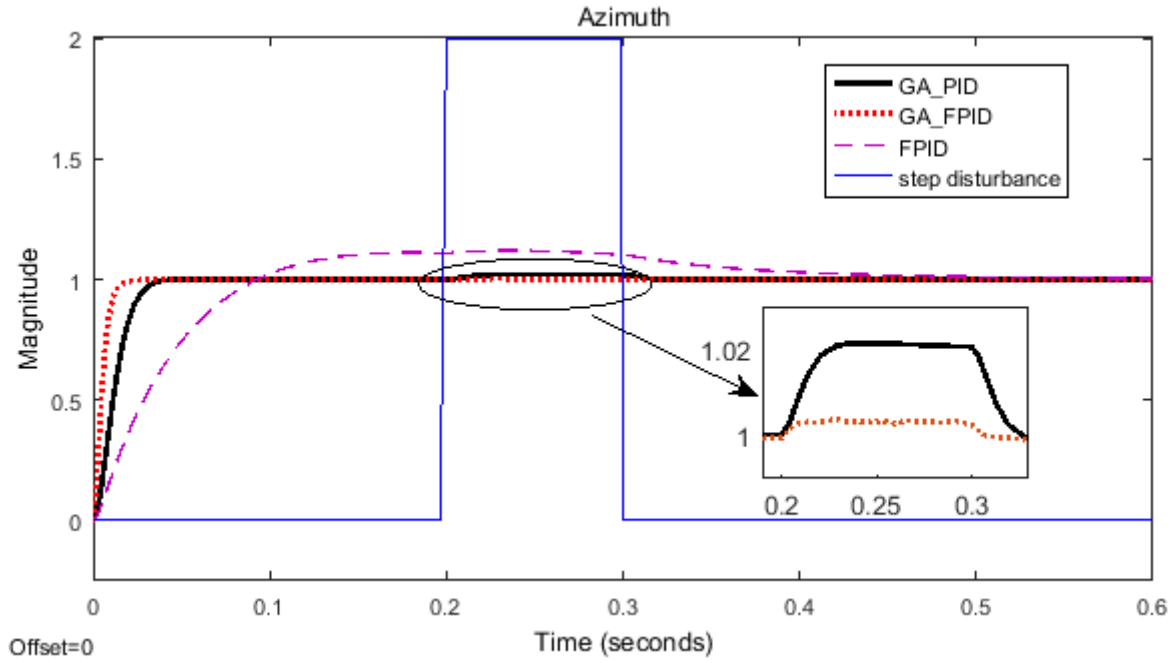


Figure 5.10: External disturbance rejection capability of the controller.

5.2 Static Camera Based Object Tracking

The interest object has been tracked with a static camera. The field of view of the camera is let fixed at some point. Interest objects are then tracked throughout the entire frame. In this case, there is a probability of missing the object of interest when it goes beyond the field of view of the camera. And that is why a dynamic camera setup is required in order to follow the path of the object by automatically adjusting the FOV.

The block diagram has been constructed in LabVIEW with vision acquisition and vision assistant tools. After configuring the camera attributes with NI-MAX, vision acquisition has been set to acquire objects in a grab mode. Then image processing techniques have been done with the vision assistant tool. While doing image processing, the interest object's kernel has been selected and for the purpose of efficiency, the gray-scale was used rather than RGB. The kernel was saved as a template and the new object has been compared to it. When it gets matched to the previously stored template, the object found button is ON, and the new captured object has been tracked. The motion of the object has been tracked by integrating the math script node and LabVIEW blocks. The front panel and plot of the trajectory of the object are given in Figure 5.11 and Figure 5.12 below. As the object gets matched to the saved template, the bounding box (highlighted in red) appeared on the object and moves synchronized with it.

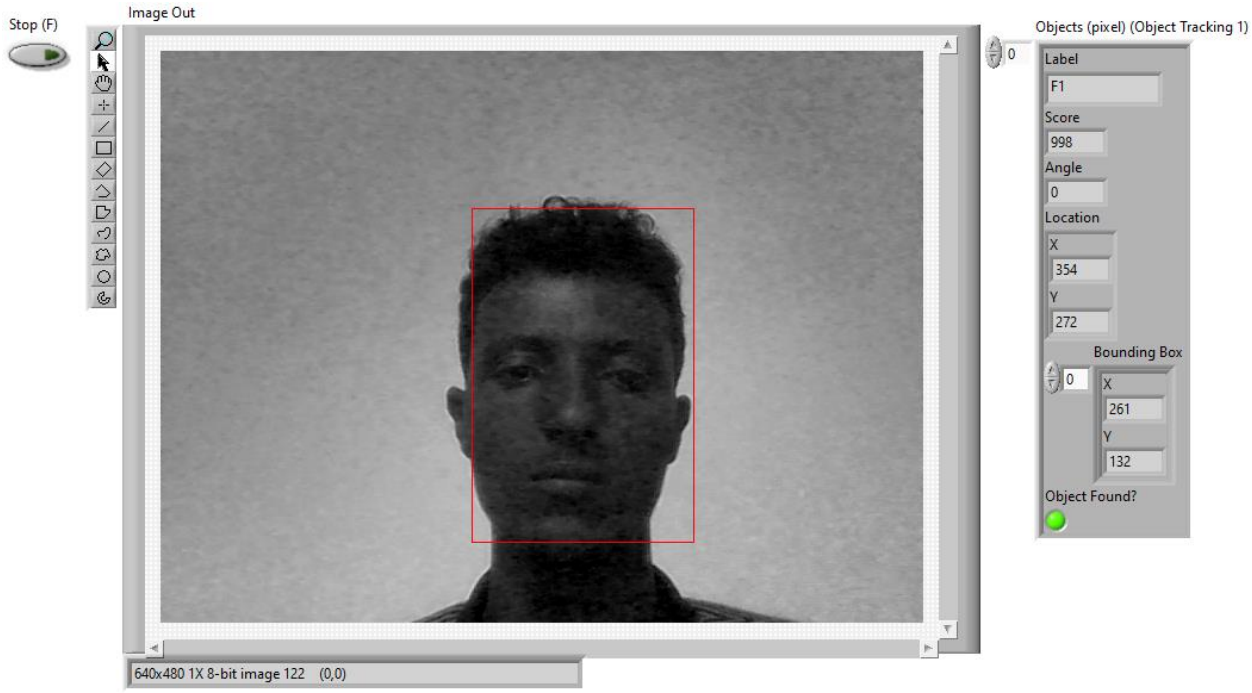


Figure 5.11: Front panel of the object tracking system.

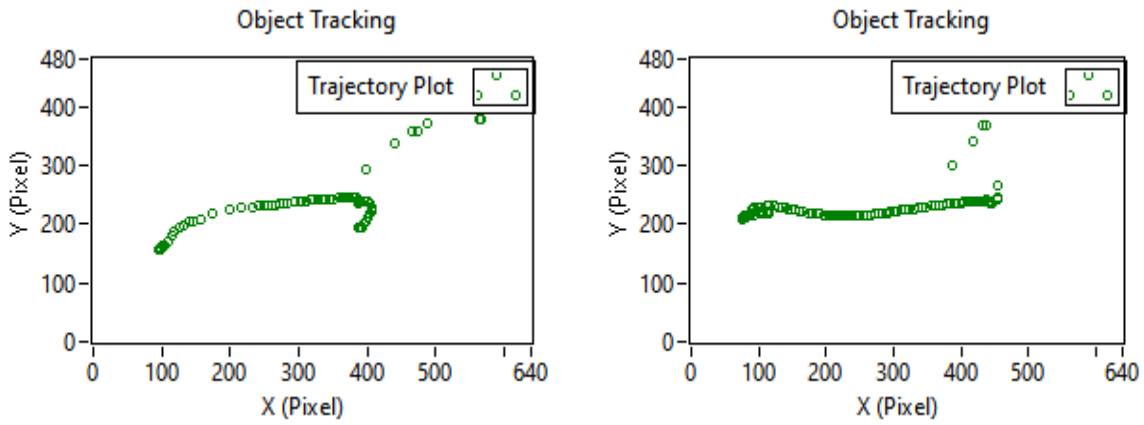


Figure 5.12: Object trajectory plots.

5.3 Dynamic Camera Based Object Tracking

As stated before, in static camera-based object tracking, the FOV is fixed and the object is lost if it goes out of the focus area of the camera. To overcome this problem the camera is then attached to a pan/tilt mechanism and let to pan and tilt in the 2-D plane. Two actuators are used to give horizontal and vertical movements to the camera. The corresponding simulation results of both the azimuth and elevation actuators are presented below.

Movement in the x Axis

Objects initial position is taken to be (0,0) and the object is let to be at the center of the camera which is (320,240).

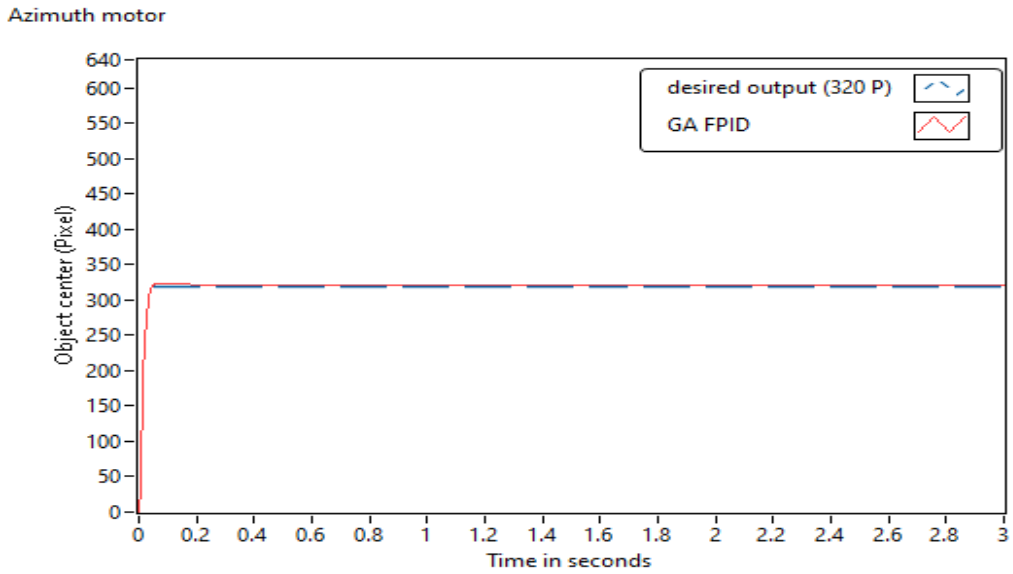


Figure 5.13: System response of the horizontal movement.

Movement in the y Axis

Similarly, the objects initial position in the y axis is taken to be 0 pixel and the set point is 240 pixels which is the center point of the captured frame.

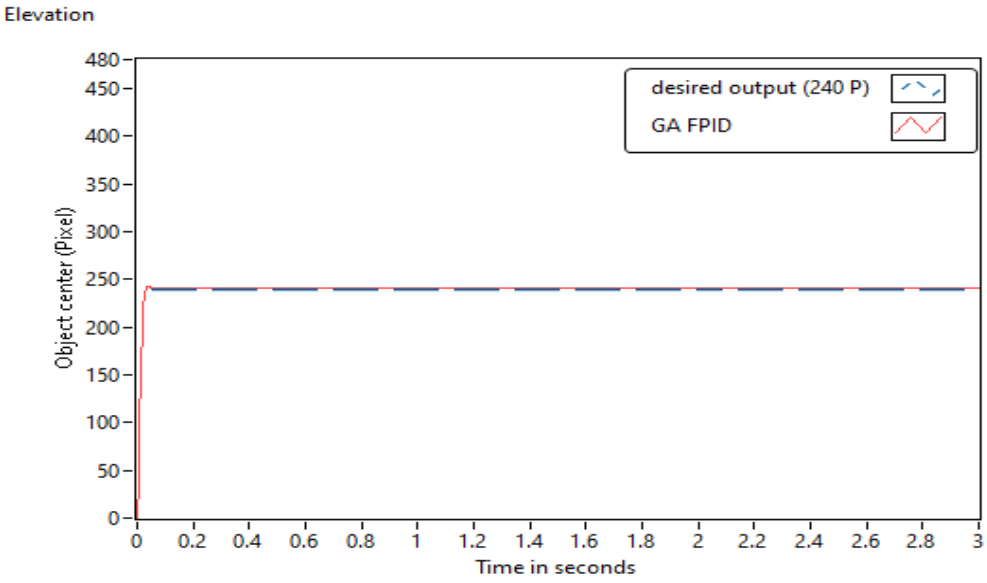
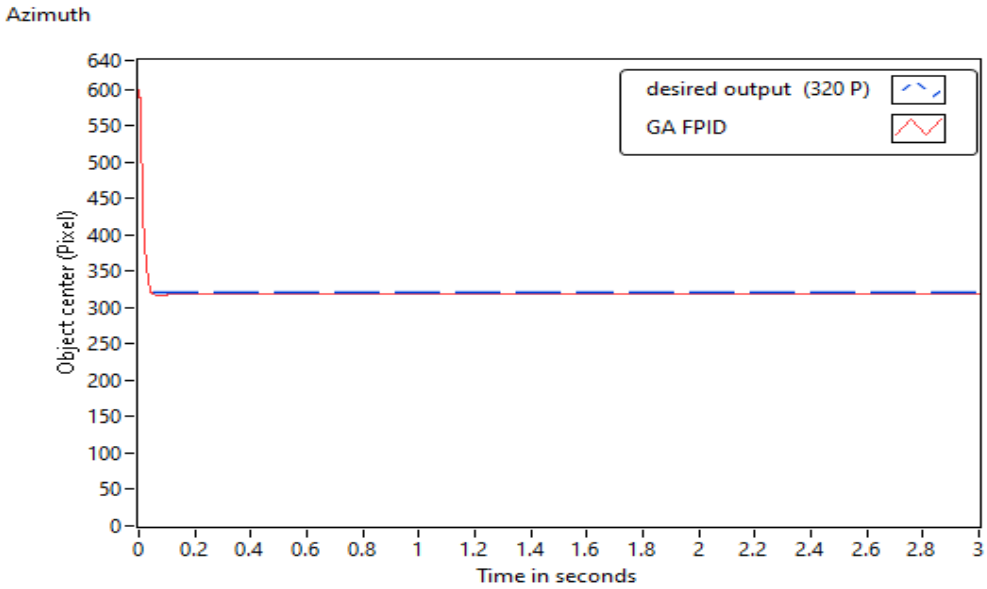
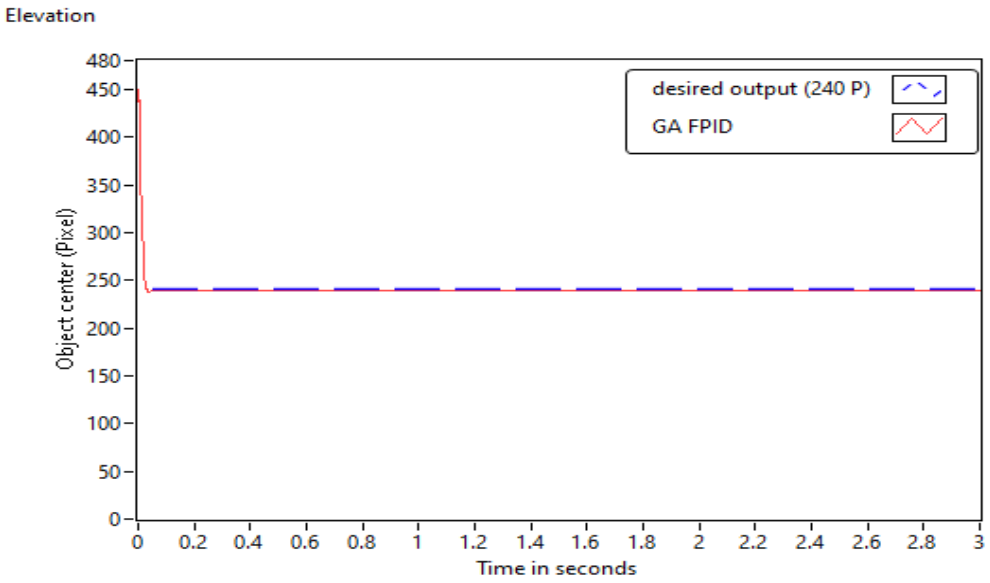


Figure 5.14: System response of the vertical movement.

As seen from the plots above, the GA-tuned FPID controller gives better results. With non-zero initial positions of the object, for example (600,450), the response plots are shown in Figures 5.15 below.



(a)

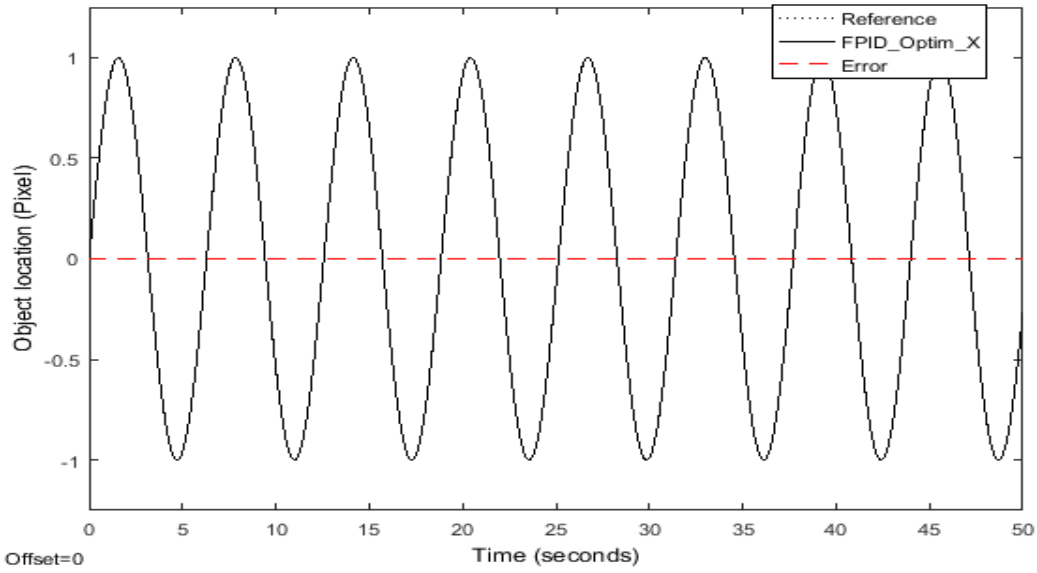


(b)

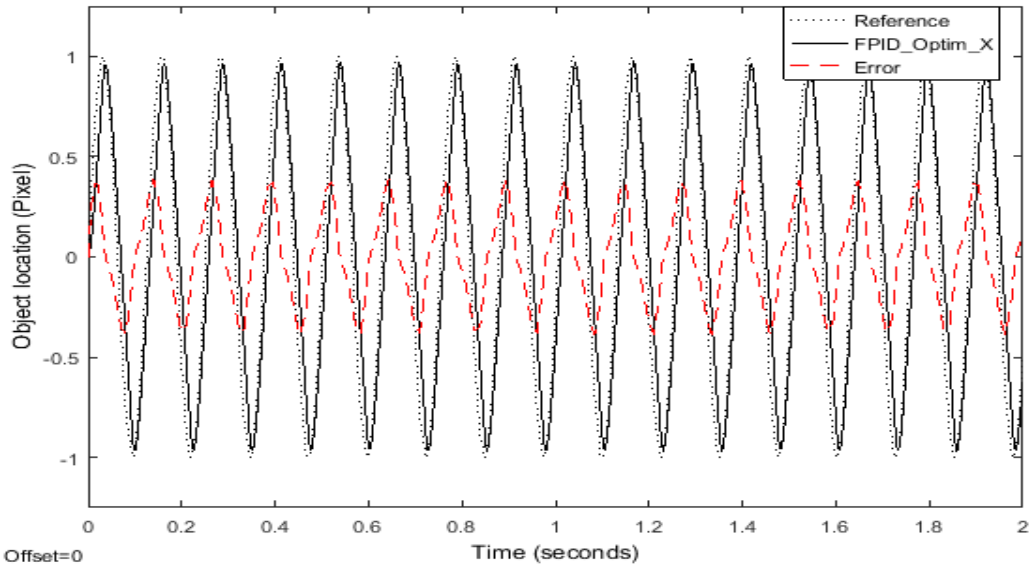
Figure 5.15: System responses with non-zero initial position for, (a): azimuth and (b): elevation.

Object Trajectory Tracking

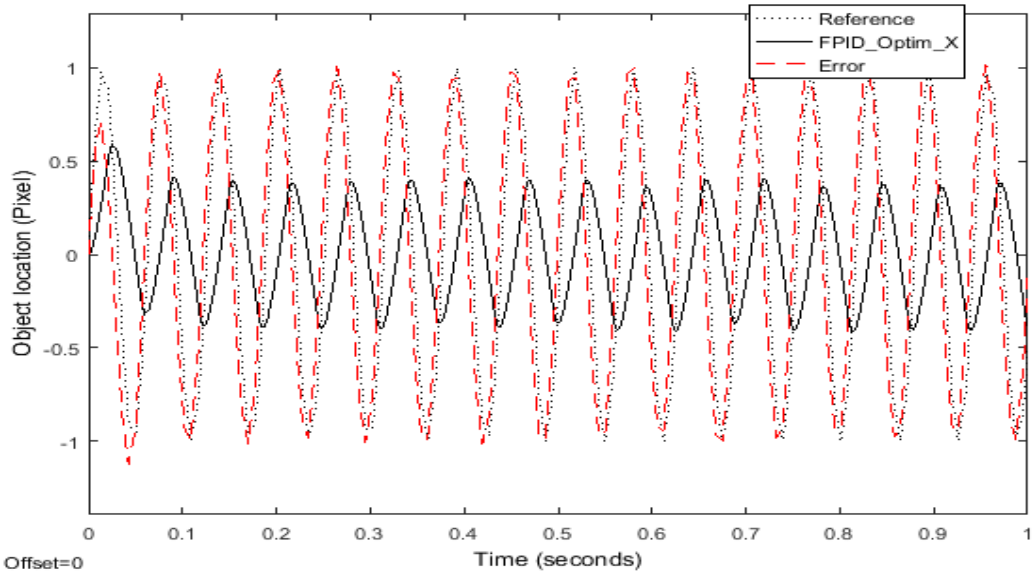
1) A sinusoidal trajectory of amplitude 1 and angular frequency 1rad/sec, 50 rad/sec and 100 rad/sec are taken as a reference signal to show the performance of the object tracking capability of the camera. The time response plots are given in Figure 5.16 below. As seen from the given plots, the sinusoidal trajectory generated is tracked with negligible steady-state error at 1 rad/sec and the steady-state error increases at high frequencies.



(a)



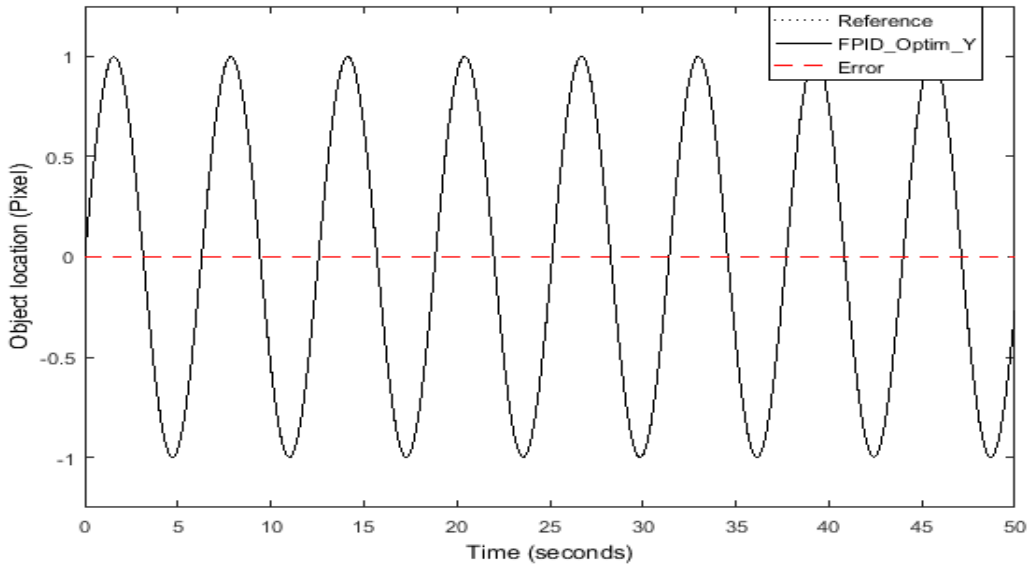
(b)



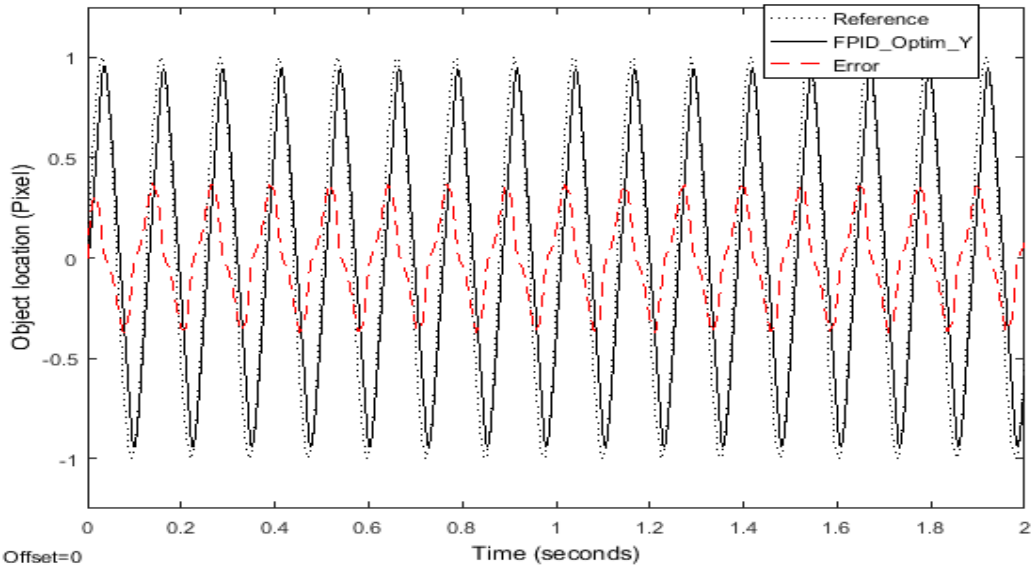
(c)

Figure 5.16: Trajectory tracking performance of the camera with GA tuned FPID Controller (a) at 1 rad/sec (b) at 50 rad/sec and (c) at 100 rad/sec for azimuth.

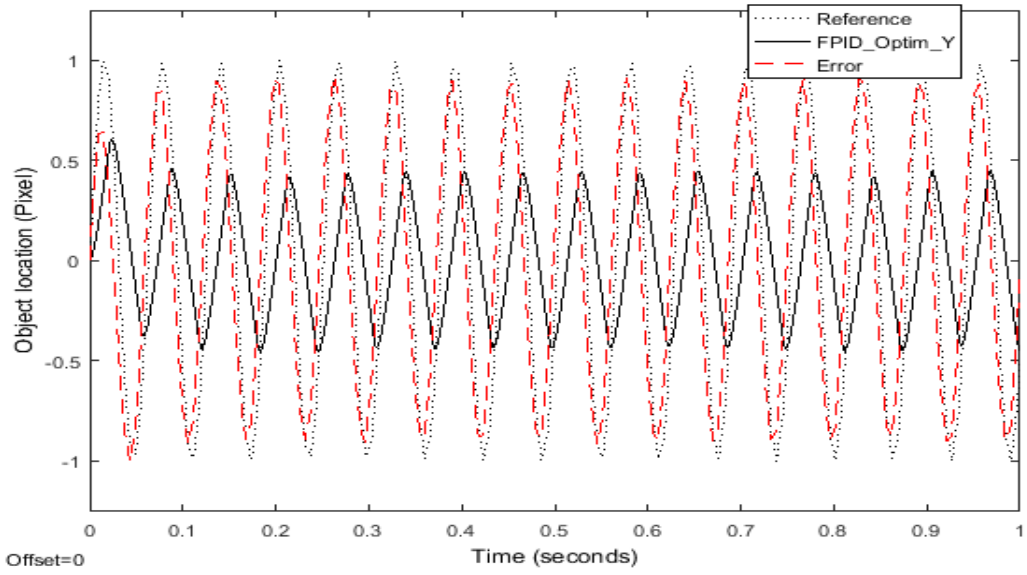
Similarly, the sinusoidal trajectory tracking performance of the elevation system is given in Figure 5.17 below. The designed controller has better response at low frequencies with acceptable tracking performance. As the tracking speed increases, the steady-state error increases and this drops the performance of the controller.


Reference
FPID_Optim_Y
Error
Object location (Pixel)
Time (seconds)

(a)


Reference
FPID_Optim_Y
Error
Object location (Pixel)
Time (seconds)
Offset=0

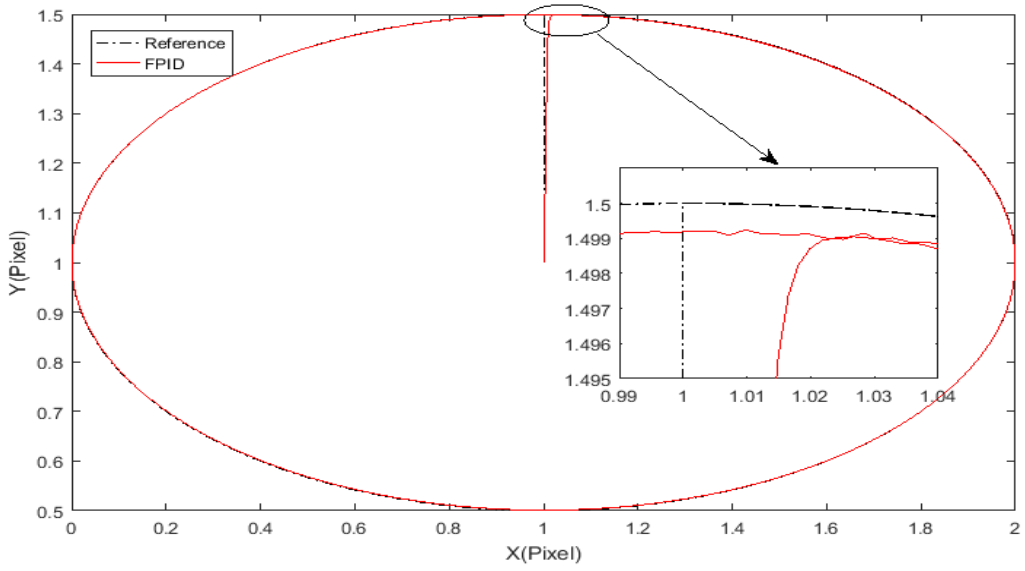
(b)



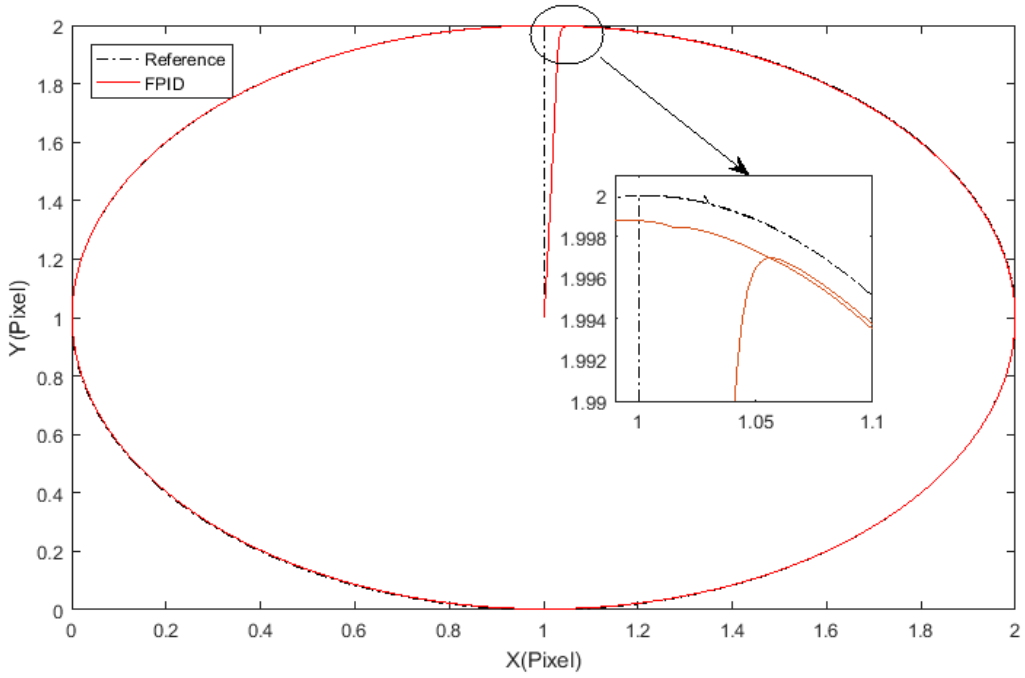
(c)

Figure 5.17: Trajectory tracking performance of the camera with GA tuned FPID Controller (a) at 1 rad/sec (b) at 50 rad/sec and (c) at 100 rad/sec for elevation.

2) Circular and elliptical trajectories are given as a reference (test) signal for the pan/tilt mechanism to demonstrate its performance of trajectory tracking capability. The angular frequency of the reference signal for the pan/tilt mechanism is taken to be 0.5 and 1 rad/sec while the magnitude is considered to be 1. As seen from Figure 5.18, the camera tracked the object with an acceptable steady-state error with GA tuned FPID controller. Figure 5.19 also shows the elliptical trajectory tracking capability of the installed system. The response showed best transient and steady-state response characteristics and best tracking performance at low frequencies.



(a)



(b)

Figure 5.18: Circular trajectory tracking capability of the pan/tilt system (a) 0.5 rad/sec and (b) 1 rad/sec.

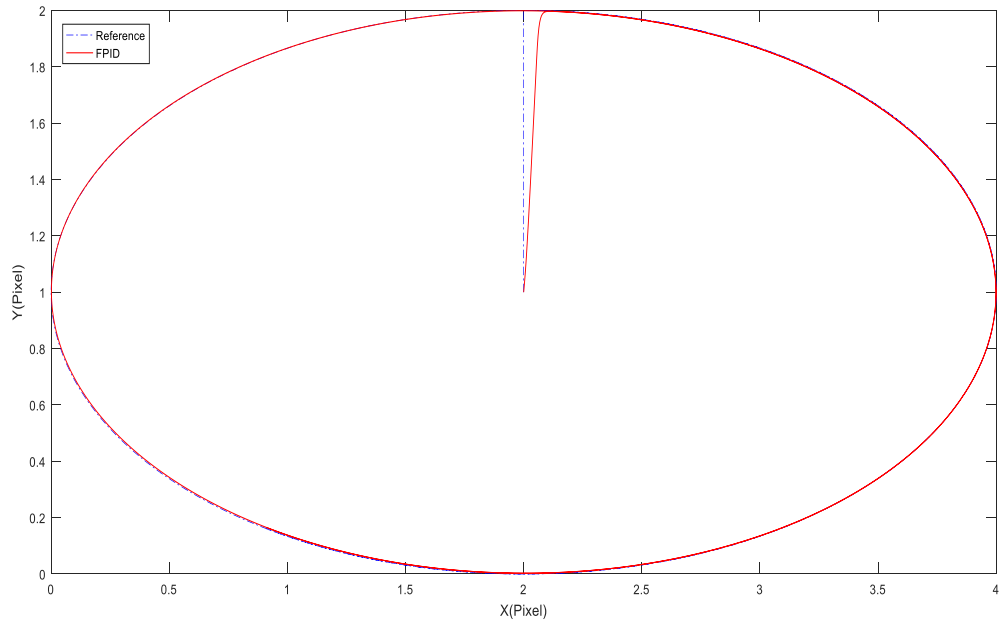


Figure 5.19: Elliptical trajectory tracking capability of the pan/tilt system at 1 rad/sec.

As the frequency increases, the controller's performance decreases which complies with the real characteristics of physical systems.

Chapter Six

Conclusion and Future Work

6.1 Conclusion

Analysis of interest object tracking has been discussed in this thesis. Both static-camera-based and dynamic-camera-based object tracking methods have been addressed. In the case of static camera-based object tracking, the object of interest's course was traced using a camera having a fixed field of view (FOV) and plotted. In this case, the camera is set fixed at some point and the object of interest was detected and tracked. As the field of view of the camera is fixed, the object is lost when it moves beyond it. The dynamic-camera-based object tracking is proposed in order to improve this problem.

In the case of dynamic camera-based object tracking, the field of view of the camera is let changing depending on the movement of the object. The camera let attached to the pan/tilt system in x and y coordinates. The camera is always directed towards the object of interest. And to effectively control the orientation of the camera, different control techniques like conventional PID, genetic algorithm tuned PID, fuzzy logic, and genetic algorithm tuned FPID controllers have been designed. As the simulation results showed, the genetic algorithm tuned FPID controller has given the best results. The corresponding rise and settling times have been reduced from 20.2 to 9.4 milliseconds and from 34 to 16 milliseconds respectively for the azimuth motor compared to the genetic algorithm tuned PID controller. Similarly, the rise and settling times reduced from 16.9 to 7.9 milliseconds and 27 to 15 milliseconds respectively for the elevation motor when compared to the genetic algorithm tuned PID controller. Maximum overshoot (%) of 0.1 and 0.2 is found for GA tuned FPID and GA tune PID controllers respectively. The steady-state errors recorded are $1.466 * 10^{-5}$ and $1.363 * 10^{-4}$ for the GA tune FPID and GA tune PID controllers respectively.

In general, this paper proposed an optimal FPID controlled design with the fuzzy variable's membership function tuned by an operating range. The GA-tuned FPID controller gave the best results considering the transient and steady-state response specifications. As seen from the results section, the actuating signals found were about 550 and 520 volts, which are impractical. To limit the controller output (actuating signals) within the actuator ratings, so as to protect the actuators, saturation has been used and the actuating signals are kept within the ratings of the

actuators. But an acceptable increase in the steady-state error has been obtained. Finally, the objective of the research to design an optimal Fuzzy-PID controller for object tracking systems has been achieved.

6.2 Future Work

Further work may focus on the implementation of an Optimal Fuzzy-PID controller for the object tracking systems to achieve an efficient and real-time object tracking system with a smart camera.

References

[1] N. V. Lopes, "Fuzzy logic-based approach for object feature tracking," Trás-os-Montes e Alto Douro University, 2012.

[2] M. Divya, A. J. I. J. o. R. Ravi Kumar, I. T. I. Computing, and Communication, "Single Object Tracking System by Using LabView," IJRITCC, vol. 4, no. 5, pp. 52-56, 2016.

[3] L. He, "Detection-assisted Object Tracking by Mobile Cameras," University of Nebraska-Lincoln, 2011.

[4] S. Ågren, "Object tracking methods and their areas of application: A meta-analysis: A thorough review and summary of commonly used object tracking methods," UMEA University, 2017.

[5] R. K. Rout, "A survey on object detection and tracking algorithms," NIoTR, India, 2013.

[6] S. Challa, M. R. Morelande, D. Mušicki, and R. J. Evans, “Fundamentals of object tracking,” Cambridge University Press, 2011.

[7] K.-S. Kwon and S. Ready, “Practical guide to machine vision software: an introduction with LabVIEW,” John Wiley & Sons, 2014.

[8] A. Yilmaz, O. Javed, and M. J. A. c. s. Shah, "Object tracking: A survey," ACM Computing Surveys, vol. 38, no. 4, pp. 13-es, 2006.

[9] E. S. Samuel, "A Neuro-Fuzzy Based Approach to Object Tracking and Motion Prediction," ed: IJSEA, 2017.

[10] G. Chen, T. T. Pham, and N. J. A. M. R. Boustany, "Introduction to fuzzy sets, fuzzy logic, and fuzzy control systems," CRC Press, vol. 54, no. 6, pp. B102-B103, 2001.

[11] P. Ponce-Cruz, A. Molina, and B. MacCleery, “Fuzzy Logic Type 1 and Type 2 Based on LabVIEW™ FPGA,” Springer, 2016.

[12] NI-LabVIEW, "PID and Fuzzy Logic Toolkit User Manual," ed: Austin-Texas, 2009.

[13] M. S. Gururaj, M. H. Ramesh, and J. A. Arvind, "A Review on Image Tracking Technique in LabVIEW," IJSDR, vol. 1, no. 6, June 2016.

[14] R. Rob, G. Tirian, and M. Panoiu, "LabVIEW application for motion tracking using USB camera," in International Conference on Applied Sciences, IOP Conf. Series: Materials Science and Engineering, 2017, vol. 200.

[15] D. Somwanshi, M. Bundele, G. Kumar, and G. J. P. C. S. Parashar, "Comparison of fuzzy-PID and PID controller for speed control of DC motor using LabVIEW," Procedia Computer Science, vol. 152, pp. 252-260, 2019.

[16] R. Firoozian, "Servo Motors and Industrial Control Theory," (in English), Springer, 2014.

[17] D. E. Roth, “Real-time multi-object tracking,” ETH Zurich, 2010.

[18] T. Klinger, “Image processing with LabVIEW and IMAQ Vision,” Prentice Hall Professional, 2003.

[19] Y. Dedeoğlu, "Moving object detection, tracking and classification for smart video surveillance," Bilkent University, 2004.

[20] F. Zanetello, "People tracking and following with a smart wheelchair using an omnidirectional camera and a RGB-D camera," University of Padua, 2014.

[21] R. Szeliski, “Computer vision: algorithms and applications,” Springer Science & Business Media, 2010.

[22] P. Rodjito, "Position tracking and motion prediction using Fuzzy Logic," Colby College, Honors Theses, 2006.

[23] M. Mehrubeoglu, L. M. Pham, H. T. Le, R. Muddu, and D. Ryu, "Real-time eye tracking using a smart camera," in 2011 IEEE Applied Imagery Pattern Recognition Workshop (AIPR), 2011, pp. 1-7: IEEE.

[24] B. Pokharel, "Machine Vision and Object Sorting: PLC Communication with LabVIEW using OPC," HAMK University of Applied Science, 2013.

[25] S. Singh, C. Shekhar, and A. J. J. o. I. Vohra, "Real-time FPGA-based object tracker with automatic pan-tilt features for smart video surveillance systems," J. Imaging, vol. 3, no. 2, p. 18, 2017.

[26] I. Lita, D. A. Visan, and I. B. Cioc, "LabVIEW application for movement detection using image acquisition and processing," in 2010 IEEE 16th International Symposium for Design and Technology in Electronic Packaging (SIITME), 2010, pp. 225-228: IEEE.

[27] J. O. J. W. T. O. S. Salim and control, "Fuzzy based PID controller for speed control of DC motor using LabVIEW," WSEAS Transactions on Systems and Control, 2015.

[28] J. Ohri, "Speed Control of DC Motor using Fuzzy Logic based on LabVIEW," IJSRP, vol. 3, no. 6, June 2013.

[29] C.-T. Chao, N. Sutarna, J.-S. Chiou, and C.-J. J. A. S. Wang, "An optimal fuzzy PID controller design based on conventional PID control and nonlinear factors," Appl. Sci., vol. 9, no. 6, p. 1224, 2019.

[30] L. Wang, “PID Control System Design and Automatic Tuning Using MATLAB/ Simulink,” John Wiley & Sons, 2020.

[31] N. I. Ali, "Simulation for position control of DC motor using fuzzy logic controller," University Tun Hussein Onn Malaysia, 2013.

[32] R. Bitter, T. Mohiuddin, and M. Nawrocki, “LabVIEW: Advanced programming techniques,” CRC press, 2017.

[33] M. J. N. I. Shiralkar, on-line: http://cnx. org/content/col10241/1.4/, Rice University, "LabVIEW Graphical Programming Course," 2007.

[34] T. LabVIEW, "LabVIEW Fundamentals," National Instruments Corporation, 2005.

[35] N. I. C., “LabVIEW Basics I Course Manual,” National Instruments Corporation, 2000.

[36] Hans-Petter Halvorsen, “Introduction to Vision Systems in LabVIEW”, University College of Southeast Norway, 2016.

[37] N. I. Corporation, “NI Vision Assistant Tutorial,” National Instruments Corporation, 2011.

[38] H. P. Halvorsen "Control Theory with Math Script Examples," University College of Southeast Norway, 2012.

[39] F. J. A. d. Haugen, "Introduction to LabVIEW Control Design Toolkit," Teach Tech, 2008.

[40] Wikipedia, the free encyclopedia, "PID controller," [Online]. Available at: https://en.wikipedia.org/wiki/PID. [Accessed 01 September 2018].

[41] Circuit Digest, “Servo motor basics, working principle and theory”. Available at: https://circuitdigest.com/article/servo-motor-basics/ Accessed: 11/25/2019 3:45 PM.

[42] Wikipedia, the free encyclopedia, "Genetic Algorithm," [Online]. Available at: https://en.wikipedia.org/wiki/Genetic_algorithm. [Accessed 16 November 2018].

[43] Towards data science, “Introduction to Genetic Algorithms-including Example Code”. Available at: https://towardsdatascience.com/introduction-to-genetic-algorithms including-example-code-e396e98d8bf3/ [Accessed 16 November 2018].

[44] Sanyo Denki, “DC Servo Systems”, Sanyo Denki Co., LTD, 2011.

[45] NI Vision, “Object Tracking Technique,” Available at: https://zone.ni.com/reference/enXX/help/372916T01/nivisionconcepts/object_tracking_techniques/ Accessed 16 November 2020.

[46] Chaitra J M, Dr Ravi Kumar A V, "Smart Autonomous Camera Tracking System Using myRIO With LabVIEW," American Journal of Engineering Research (AJER), vol. 7, no. 5, pp. 408-413, 2018.

[47] A. Noshadi, J. Shi, W. S. Lee, P. Shi, and A. Kalam, "Optimal PID-type fuzzy logic controller for a multi-input multi-output active magnetic bearing system," Neural computing and applications, In Press, DOI: 10.1007/s00521-015-1996-7. vol. 27, no. 7, pp. 2031-2046, 2016.

[48] R. J. A.-R. E. J. Ilham Majeed, "Design and FPGA implementation of Takagi-Sugeno fuzzy controller Based on LUTs," vol. 18, no. 6, pp. 81-94, 2010.

Appendix A

A.1 GA Tuned Fuzzy-PID Controller Block Diagram for Pan/Tilt Mechanism

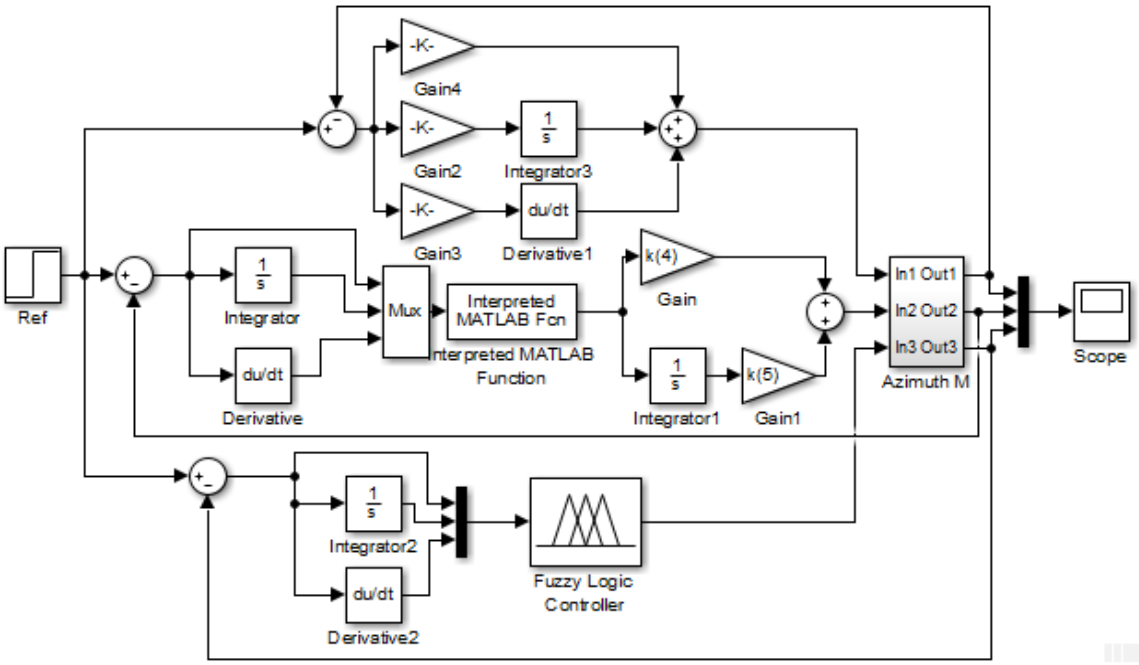


Figure A.1: Block diagram of azimuth motor PID and FPID controller design in Simulink.

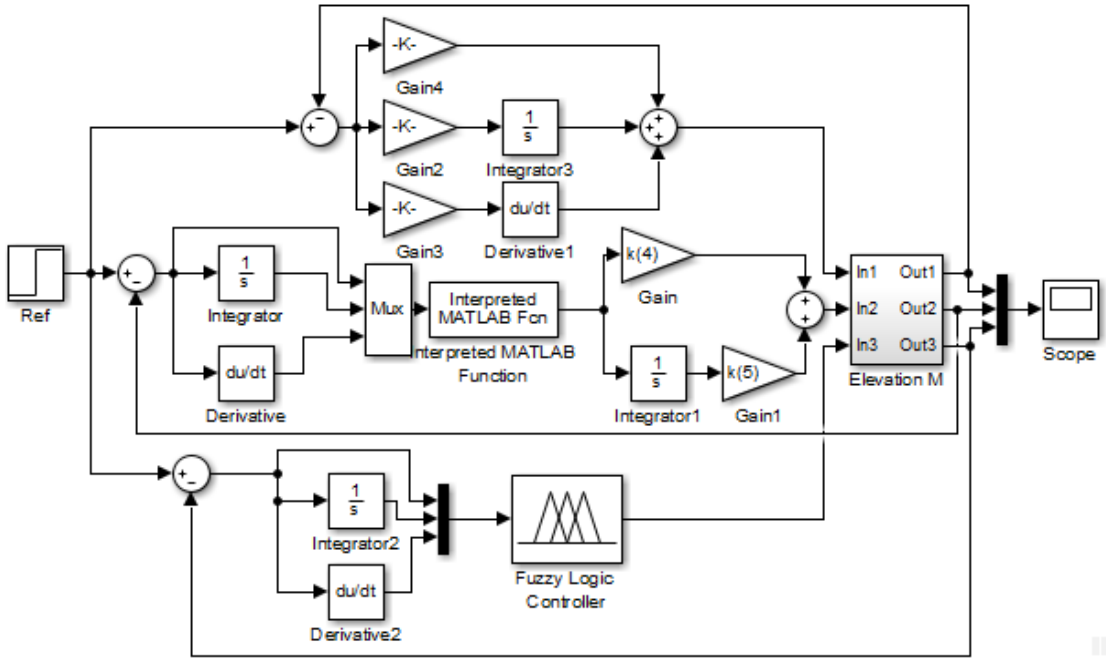


Figure A.2: Block diagram of elevation motor PID and FPID controller design in Simulink.

A.2 The Corresponding Optimization Tool GUI and Fitness Function Value

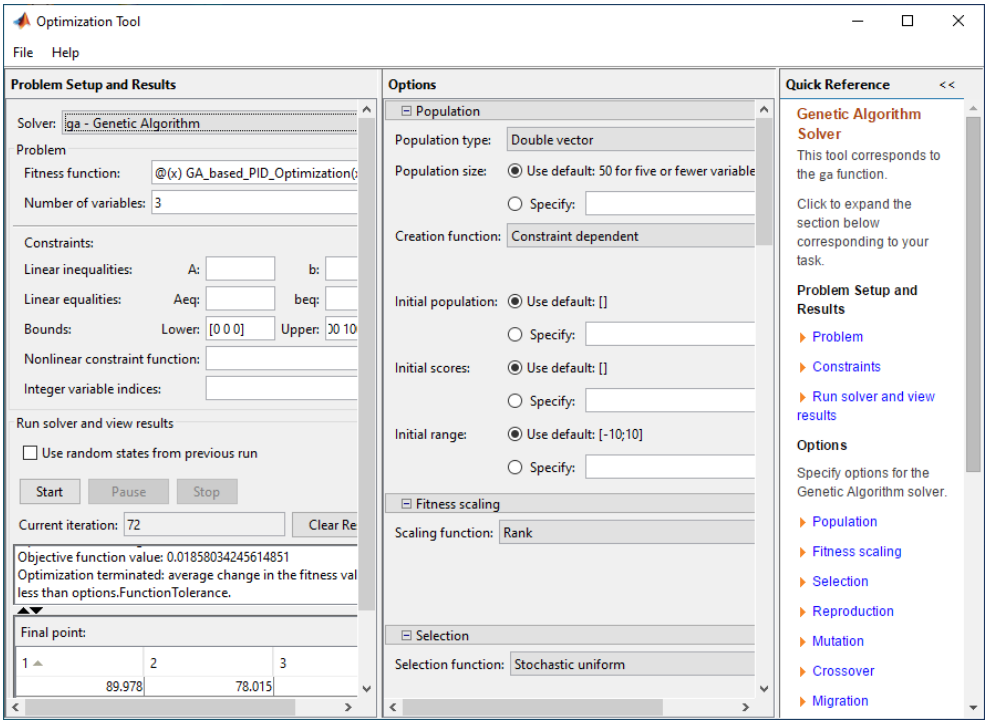


Figure A.3: MATLAB optimization tool box.

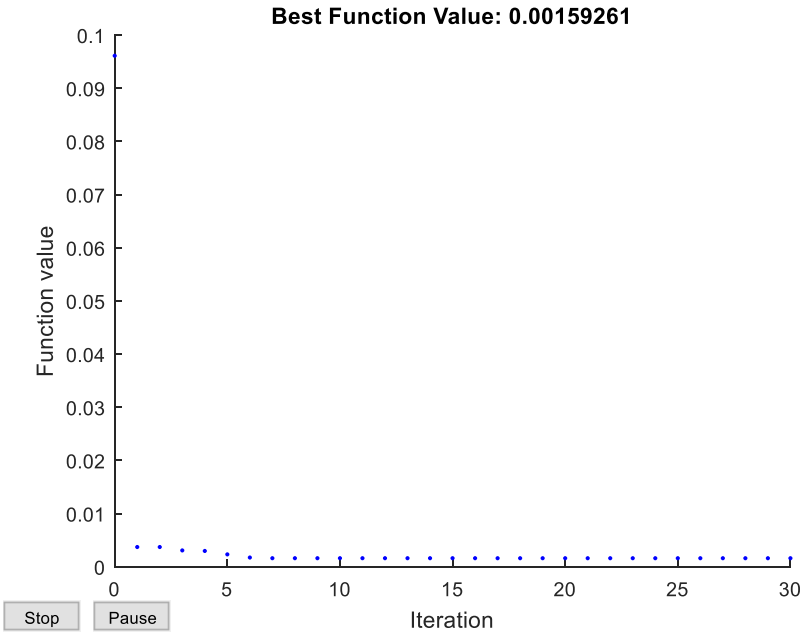


Figure A.4: FPID optimization fitness function.

A.3 MATLAB Code for GA Based PID Optimization

```
% ---------------------------------------------------------------------
%  Genetic algorithm code development for PID Optimization
```

```
%  This code with optimtool tool box generates the optimized PID controller
%  parameter values
%-------------------------------------------------------------------------
function [J]=GA_based_PID_Optimization(x)
lb=[0 0 0]; ub=[100 100 100];
assignin('base', 'x', x);
sim('GA_based_PID_Optim.slx');
J=ITAE(length(ITAE)); end
```

A.4 MATLAB Code for GA Based FPID Optimization

```
% -------------------------------------------------------------------------
%  Genetic algorithm code development for FPID Optimization. This code in
%  line with optimization tool box generates the optimized FPID controller
%  parameter values
%-------------------------------------------------------------------------
function [J]=FPID_GA_Optim(k)
lb=[0 0 0 0 0]; ub=[500 5000 5000 5000 5000];
assignin('base', 'k', k);
%sim('FPID_Optimization_X.slx',0.01); % for Azimuth
sim('FPID_Optimization_Y.slx',0.01);  % for Elevation
J=ITAE(length(ITAE)); end
function out=FPID_XM(input)
k=evalin('base','k'); warning off;
error=input(1); error_int=input(2); error_dot=input(3);
tfc=newfis('tfc','sugeno'); % Sugeno type FLC
sa1=[0.5,0.5]; % Membership function distribution for error
sb1=[0.5,0.5]; % Membership function distribution for integral error
sc1=[0.5,0.5]; % Membership function distribution for derivative error
sd1=[0.167,0.167,0.167,0.167,0.167];% MF distribution for output
% Input and output declaration
tfc=addvar(tfc,'input','error',k(1)*[-1 1]);
tfc=addvar(tfc,'input','error_int',k(2)*[-1 1]);
tfc=addvar(tfc,'input','error_dot',k(3)*[-1 1]);
tfc=addvar(tfc,'output','u1',[-1 1]);
% Membership function for input 1 (error)
tfc=addmf(tfc,'input',1,'NB','trimf',k(1)*[-1 -1 -sa1(2)]);
tfc=addmf(tfc,'input',1,'NS','trimf',k(1)*[-1 -sa1(1) 0]);
tfc=addmf(tfc,'input',1,'ZO','trimf',k(1)*[-sa1(1) 0 sa1(1)]);
tfc=addmf(tfc,'input',1,'PS','trimf',k(1)*[0 sa1(1) 1]);
tfc=addmf(tfc,'input',1,'PB','trimf',k(1)*[sa1(2) 1 1]);
```

```
% Membership function for input 2 (error_int)
tfc=addmf(tfc,'input',2,'NB','trimf',k(2)*[-1 -1 -sb1(2)]);
tfc=addmf(tfc,'input',2,'NS','trimf',k(2)*[-1 -sb1(1) 0]);
tfc=addmf(tfc,'input',2,'ZO','trimf',k(2)*[-sb1(1) 0 sb1(1)]);
tfc=addmf(tfc,'input',2,'PS','trimf',k(2)*[0 sb1(1) 1]);
tfc=addmf(tfc,'input',2,'PB','trimf',k(2)*[sb1(2) 1 1]);
% Membership function for input 3 (error_dot)
tfc=addmf(tfc,'input',3,'NB','trimf',k(3)*[-1 -1 -sc1(2)]);
tfc=addmf(tfc,'input',3,'NS','trimf',k(3)*[-1 -sc1(1) 0]);
tfc=addmf(tfc,'input',3,'ZO','trimf',k(3)*[-sc1(1) 0 sc1(1)]);
tfc=addmf(tfc,'input',3,'PS','trimf',k(3)*[0 sc1(1) 1]);
tfc=addmf(tfc,'input',3,'PB','trimf',k(3)*[sc1(2) 1 1]);
% Membership function for output (u)
tfc=addmf(tfc,'output',1,'NE6','constant',[-1]);
tfc=addmf(tfc,'output',1,'NE5','constant',[-sd1(5)]);
tfc=addmf(tfc,'output',1,'NE4','constant',[-sd1(4)]);
tfc=addmf(tfc,'output',1,'NE3','constant',[-sd1(3)]);
tfc=addmf(tfc,'output',1,'NE2','constant',[-sd1(2)]);
tfc=addmf(tfc,'output',1,'NE1','constant',[-sd1(1)]);
tfc=addmf(tfc,'output',1,'NE0','constant',[0]);
tfc=addmf(tfc,'output',1,'PE1','constant',[sd1(1)]);
tfc=addmf(tfc,'output',1,'PE2','constant',[sd1(2)]);
tfc=addmf(tfc,'output',1,'PE3','constant',[sd1(3)]);
tfc=addmf(tfc,'output',1,'PE4','constant',[sd1(4)]);
tfc=addmf(tfc,'output',1,'PE5','constant',[sd1(5)]);
tfc=addmf(tfc,'output',1,'PE6','constant',[1]);
% Rule Matrix
rule_matrix = [  1  1  1  1  1  1;
                 1  2  1  2  1  1;
                 1  3  1  3  1  1;
                 1  4  1  4  1  1;
                 1  5  1  5  1  1;
                 2  1  1  2  1  1;
                 2  2  1  3  1  1;
                 2  3  1  4  1  1;
                 2  4  1  5  1  1;
                 2  5  1  6  1  1;
                 3  1  1  3  1  1;
                 3  2  1  4  1  1;
```

```
                .
                .
                .
        5   3   4   10   1   1;
        5   4   4   11   1   1;
        5   5   4   12   1   1;
        1   1   5   5   1   1;
        1   2   5   6   1   1;
        1   3   5   7   1   1;
        1   4   5   8   1   1;
        1   5   5   9   1   1;
        2   1   5   6   1   1;
        2   2   5   7   1   1;
        2   3   5   8   1   1;
        2   4   5   9   1   1;
        2   5   5   10   1   1;
        3   1   5   7   1   1;
        3   2   5   8   1   1;
        3   3   5   9   1   1;
        3   4   5   10   1   1;
        3   5   5   11   1   1;
        4   1   5   8   1   1;
        4   2   5   9   1   1;
        4   3   5   10   1   1;
        4   4   5   11   1   1;
        4   5   5   12   1   1;
        5   1   5   9   1   1;
        5   2   5   10   1   1;
        5   3   5   11   1   1;
        5   4   5   12   1   1;
        5   5   5   13   1   1;
];
tfc=addrule(tfc,rule_matrix);
out= evalfis([error error_int error_dot],tfc);
```