Speed Control of Induction Motor using Fuzzy Logic Approach

A PROJECT THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF

Bachelor of Technology
In
Electrical Engineering

By

Varuneet Varun
(Roll 108EE011)

G. Bhargavi
(Roll 108EE026)

Suneet Nayak
(Roll 108EE044)

Under Supervision of
Prof. Kanungo Barada Mohanty

Department of Electrical Engineering
National Institute of Technology, Rourkela
Rourkela- 769008, Odisha

National Institute of Technology
Rourkela

Certificate

This is to certify that the work contained in this thesis, titled **"SPEED CONTROL OF INDUCTION MOTOR USING FUZZY LOGIC APPROACH"** submitted by **Varuneet Varun, G. Bhargavi** and **Suneet Nayak** is an authentic work that has been carried out by them under my supervision and guidance in partial fulfillment for the requirement for the award of Bachelor of Technology Degree in Electrical Engineering at National Institute of Technology, Rourkela.

To the best of my knowledge, the matter embodied in the thesis has not been submitted to any other University/ Institute for the award of any Degree or Diploma.

Place: Rourkela

Date: 11th May, 2012

Dr. Kanungo Barada Mohanty

Associate Professor
Department of Electrical Engineering
National Institute of Technology
Rourkela – 769008

Acknowledgment

We are grateful to **The Department of Electrical Engineering** for giving us the opportunity to carry out this project, which is an integral fragment of the curriculum in B. Tech programme at the National Institute of Technology, Rourkela.

We would like to express our heartfelt gratitude and regards to our project guide, **Prof. Dr. K. B. Mohanty**, Department of Electrical Engineering, for being the corner stone of our project. It was his incessant motivation and guidance during periods of doubts and uncertainties that has helped us to carry on with this project.

We would like to thank **Prof. (Dr.) B. D. Subudhi**, Head of the Department, Electrical Engineering for his guidance, support and direction. We are also obliged to the staff of Electrical Engineering Department for aiding us during the course of our project. We would also like to offer our sincere thanks to Prof. (Dr.) Subhojit Ghosh and Mr. Y. Suresh from the Department of Electrical Engineering, for their continual guidance in programming. We offer our heartiest thanks to our friends for their help in collection of data samples whenever necessary.

Last but not the least, we want to acknowledge the contributions of our parents and family members, for their constant and never ending motivation.

VARUNEET VARUN (108EE011)

G.BHARGAVI (108EE026)

SUNEET NAYAK (108EE044)

Abstract

This thesis presents a methodology for implementation of a rule-based fuzzy logic controller applied to a closed loop Volts/Hz induction motor speed control. The Induction motor is modeled using a dq axis theory. The designed Fuzzy Logic Controller's performance is weighed against with that of a PI controller. The pros of the Fuzzy Logic Controllers (FLCs) over the conventional controllers are: (i) they are economically advantageous to develop, (ii) a wider range of operating conditions can be covered using FLCs, and (iii) they are easier to adapt in terms of natural language. Another advantage is that, an initial approximate set of fuzzy rules can be impulsively refined by a self-organizing fuzzy controller. For V/f speed control of the induction motor, a reference speed has been used and the control architecture includes some rules. These rules portray a nonchalant relationship between two inputs and an output, all of which are nothing but normalized voltages. These are:

- The input speed error denoted by Error (**e**).
- The input derivative of speed error denoted by Change of error (**Δe)**, and
- The output frequency denoted by Change of Control ***(ω_{sl})***.

The errors are evaluated according to the rules in accordance to the defined member functions. The member functions and the rules have been defined using the FIS editor given in MATLAB. Based on the rules the surface view of the control has been recorded. The system has been simulated in MATLAB/SIMULINK® and the results have been attached. The results obtained by using a conventional PI controller and the designed Fuzzy Logic Controller has been studied and compared. The controller has then been tuned by trial and error method and simulations have been run using the tuned controller.

Keywords : *V/f induction motor speed control, dq axis theory, Fuzzy Logic controller, Mamdani Architecture*

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

"The purpose of computing is insight, not numbers!"

(R. W. Hamming)

1.1 Introduction

The use of induction motors has increased tremendously since the day of its invention. They are being used as actuators in various industrial processes, robotics, house appliances (generally single phase) and other similar applications. The reason for its day by day increasing popularity can be primarily attributed to its robust construction, simplicity in design and cost effectiveness. These have also proved to be more reliable than DC motors. Apart from these advantages, they have some unfavorable features like their time varying and non-linear dynamics.

Speed control is one of the various application imposed constraints for the choice of a motor. Hence, in the last few years it has been studied by many, and various methods for the same have been developed. An insight into the same has been provided in Chapter 2. Out of all the speed control mechanisms, the Volts/Hertz control scheme is very popular because it provides a wide range of speed control with good running and transient performance. This scheme has been thoroughly explained in Chapter 2. This control mechanism is referred to as scalar control mode. Here both the input and output commands are speed, unlike the Vector control mode where it is torque/flux and reference current, respectively. Even though vector control drives provide excellent performance in terms of dynamic speed regulation, implementation of the same is tedious owing to on-line coordinate transformations that convert line currents into two axis representation and vice versa [8].

The field of power electronics has contributed immensely in the form of voltage-frequency converters which has made it possible to vary the speed over a wide range [11] [13]. However, the highly non-linear nature of the induction motor control dynamics demands strenuous control algorithms for the control of speed. The conventional controller types that are used for the aforementioned purpose are may be numeric or neural or fuzzy. The controller types that are regularly used are: Proportional Integral (PI), Proportional Derivative (PD), Proportional Integral Derivative (PID), Fuzzy Logic Controller (FLC) or a blend between them.

The PID controller offers a very efficient solution to numerous control problems in the real world. If PID controllers are tuned properly [1] [2], they can provide a robust and reliable control. This very feature has made PID controllers exceedingly popular in industrial applications. The only problem associated with use of conventional PI, PD and PID controllers in speed control of induction motors is the complexity in design arising due to the non-linearity of Induction Motor dynamics. The conventional controllers have to linearize the non-linear systems in order to calculate the parameters [10]. To obtain a perfect non-linear model is almost impossible and hence the values of the parameters that are obtained from it are thereby approximate. Again, Variable Speed Drives (VSD) for Induction Motor (IM) require wide operating speed range along with a fast torque response, irrespective of the variations in load, thereby leading us towards more advanced methods of control so as to meet the real demand. The conventional control methods possess the following difficulties:

- Dependence on the exactness of the mathematical model of the system.
- Expected performance not being met due to the load disturbance, motor saturation and thermal deviations.
- Decent performance exhibited only at one operating speed when classical linear control is employed.
- Adopting the right coefficients for acceptable results.

From the above, it can be deduced that in order to implement conventional control methodologies, it is necessary to have knowledge of the system's model that is to be controlled. The usual method of computation of mathematical model of the induction motor is difficult, due to the non-linearity of motor dynamics. Whenever a variation in system or ambient parameters arises, the system's behaviour becomes non-pleasing. The conventional controllers designed to provide high performance increase the design complexity along with the cost. Of late, soft computation techniques are being used widely in induction motor drives (and, in general all other electrical drives) and they are:

- 1. Artificial Neural Network (ANN)
- 2. Fuzzy Logic Set (FLS)
- 3. Fuzzy-Neural Network (FNN)
- 4. Genetic Algorithm Based system (GAB)
- 5. Genetic Algorithm Assisted system (GAA)

Thus, to overcome the complexities of conventional controllers, fuzzy control [3] has been implemented in many motor control applications. In the last three decades, fuzzy control has gained much popularity owing to its knowledge based algorithm, better non-linearity handling features and independence of plant modeling. The Fuzzy Logic Controller (FLC) owes its popularity to linguistic control. Here, an exact mathematical model for the system to be controlled is not required [4]. Hence, Fuzzy logic basically tries to replicate the human thought process in its control algorithm. The FLC has thereby proven to be very beneficial in the industries as it has the proficiency to provide complex non-linear control to even the uncertain nonlinear systems. In addition to the aforementioned attributes, a fuzzy logic controller also makes good performance in terms of stability, precision, reliability and rapidity achievable.

It takes two inputs, viz. error and rate of change of error to model a FLC with the help of simple if-then rules. No complicated hardware is required for the same. The modeling of an FLC has been explained in Chapter 4.

1.2 Advantages of Fuzzy Logic Controller

The advantages provided by a FLC are listed below:

- It is simple to design.
- It provides a hint of human intelligence to the controller.
- It is cost effective.

- No mathematical modeling of the system is required.
- Linguistic variables are used instead of numerical ones.
- Non-linearity of the system can be handled easily.
- System response is fast.
- Reliability of the system is increased.
- High degree of precision is achieved.

These advantages allow fuzzy controllers can be used in systems where description of the process and identification of the process parameters with precision is highly difficult. Hence it provides a fuzzy characteristic to the control mechanism.

1.3 Project Objective

The main objective of this project is to develop a fuzzy logic based controller to control the speed of the induction motor, employing the scalar control model. The voltage and frequency input to the induction motor are to be controlled in order to obtain the desired speed response.

1.4 Scope of the Project

The scope of the project is:

1) Development an artificially intelligent speed controller using the fuzzy logic approach and based on the scalar control model.
2) To produce a learning package of Fuzzy Logic Controller, for scalar speed control of Induction Motor, for future reference purpose.

1.5 Organization of the Project Report

This documentation deals with the proposed idea of a fuzzy controller for a closed loop speed control of an induction motor with volts/ Hz relation as the parameter. In other words, scalar control method is used for the speed control of the Induction Motor.

This report is divided into eight chapters. Chapter 1 is an introduction and gives an overview of the project and speaks about the scope and the main objective.

Chapter 2 discusses briefly about the induction motor and scalar speed control or V/f control of the induction motor. It also provides an insight into induction motor modeling.

Chapter 3 gives an overview of the fuzzy logic. It discusses about the fuzzy sets, their operation and membership functions.

Chapter 4 provides the basic information about Fuzzy Logic Controllers (FLC), its various features and their functioning.

Chapter 5 deals with the design of the controller based on the Mamdani architecture. The block diagram that will be used to control the induction motor is included in this chapter. This chapter also encompasses the controller design with the help of MATLAB and the various steps used for the same.

Chapter 6 is dedicated to the simulation of the induction motor speed control system in MATLAB/SIMULINK®. Both Fuzzy Logic Controller and conventional PI Controller have been used. The results obtained have been compared and discussed.

In Chapter 7 the previously designed Fuzzy Logic Controller is tuned so as to remove the flaws that were found in the controller's performance.

This is followed by the conclusion in Chapter 8. This chapter also includes information about the future scope of the designed controller.

CHAPTER 2

Induction Motor Drives

"It is possible to fly without motors, but not without knowledge and skill."

(Wilbur Wright)

2.1 Introduction

The induction motor finds its place amongst more than 85% of industrial motors as well as in its single-phase form in various domestic usages. Markedly a constant-speed motor with shunt characteristic, speed drops only by a few percent from no-load to full load. Hence in the past, induction motors have been used primarily in constant speed applications. Traditional methodologies employing speed control have either been high-priced or very inefficient, unlike the dc motor. Nonetheless, the presence of commutator and brushes in the latter, which require recurrent maintenance make dc motor drives improper for use in hazardous and polluted environments. On the other hand, owing to the simple, rugged, cheaper, smaller and subsequently lighter build of induction motor drives (particularly squirrel-cage type), they are designed for fans, blowers, cranes, traction, conveyers, etc. in spite of finding stiff competition from dc drives for such applications.

2.2 Construction and Operation

Like any electric motor, a 3-phase induction motor has a stator and rotor. The stator of an induction motor, wound for three phases and a definite number of poles, is made up of a number of stampings with evenly spaced slots to carry the three phase winding. **Greater is the number of poles, lesser is the speed of the motor and vice-versa.** Upon energizing the stator winding, a rotating magnetic field of constant magnitude is produced, and the rotor winding derives currents from the peripheral stator through electromagnetic induction and hence the name. Two types of construction are utilized for the rotor – *wound-rotor* and *squirrel-cage rotor* [13]. It is mounted on a shaft and is a hollow laminated core having slots on its outer edge.

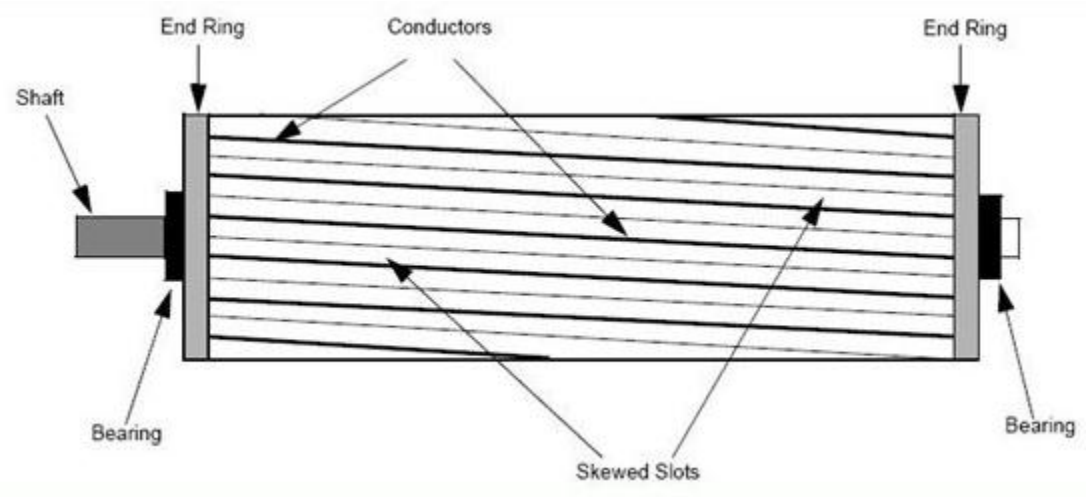


Fig 1: Squirrel Cage Rotor construction

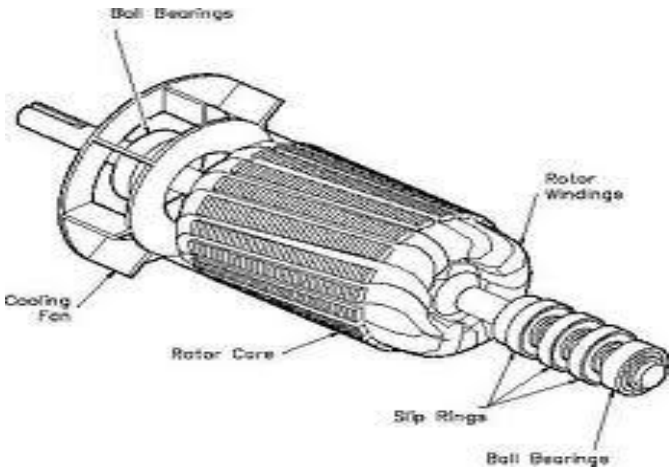


Fig 2: Wound Rotor construction

2.2.1 Principle of rotating magnetic field

When a three phase voltage is applied to the stator winding, a rotating magnetic field is produced. It is called a rotating field since its poles do not remain in a fixed position on the stator but go on shifting their positions surrounding the stator. The magnitude of this field is constant and equal to $1.5\phi_m$, where ϕ_m is the maximum flux due to any phase.

On energizing the three phase stator from a three phase supply, a rotating magnetic field sets up round the stator which rotates at synchronous speed n_S. This field passes through the air-gap and cuts the stationary rotor conductors. Owing to the relative speed between the rotating flux and the static rotor, electromotive forces are induced in the rotor conductors. For the reason that the rotor circuit is short-circuited, currents start flowing in the rotor conductors. Again, these conductors are placed in the magnetic field produced by the stator. As a result, mechanical force acts on the rotor conductors. A torque, produced as a result of this force, tends to move the rotor in the same direction as the rotating field. This is justified by Lenz's law, according to which the direction of rotor currents will be such that they have a tendency to oppose the cause producing them. Now, the relative speed between the rotating field and the standstill rotor conductors is the cause generating the rotor currents. Thus to reduce this speed, the rotor starts running in the same direction as that of stator field and tries to catch it. Clearly, the rotor speed n is always less than the stator field speed n_S [13].

2.3 Speed Control of Induction Motors

The speed control of induction motors involves more complicacy than the control of dc motor, especially if comparable accuracy is desired. The main reason for the same can be attributed to the complexity of the mathematical model of the induction machine, as well as the complicated power converters supplying this motor. Variable speed induction motor drives employ various control algorithms.

2.3.1 Speed Regulation as a Means of Controlling a Process

Let us consider the process of driving to work. Driving at the highest possible speed would probably cause an accident. And driving at a single speed that will be safe for every portion of the route will take long to reach to the destination. Hence adjusting the speed which goes well with the route minimizes the time to accomplish the objective of the process within limits of reliable operation.

The process control benefits that may be provided by an adjustable speed drive are as follows:

- Smoother operation.
- Acceleration control as an added incentive.
- Varying operating speed for each process.
- Compensates for fluctuating process parameters.
- Permits slow operation for setup purpose.
- Allows accurate positioning.
- Provides torque control.

2.3.2 Speed Control Techniques

Mathematically, the relation between the speed of an induction motor and the synchronous speed (i.e., the speed at which the revolving flux rotates) can be stated as

$$n = (1 - s)n_S \tag{2.1}$$

Also,
$$n_S = \frac{120f}{P} \tag{2.2}$$

which implies that there are two basic ways of speed control, namely

(i) Slip-control for fixed synchronous speed.
(ii) Control of synchronous speed.

A stricter sorting reveals the following methods [13]:

(i) Pole changing.
(ii) Stator voltage control.
(iii) Supply frequency control.
(iv) Eddy-current coupling.
(v) Rotor resistance control.
(vi) Slip power recovery.

2.3.3 Variable Frequency Control from Voltage Sources

Synchronous speed (and hence the motor speed) can be controlled by varying supply frequency.

Voltage induced in stator $E_1 \propto \phi f$, where ϕ is the air-gap flux and f is the supply frequency.

Neglecting the stator voltage drop (which is hardly 10% of the supply voltage),

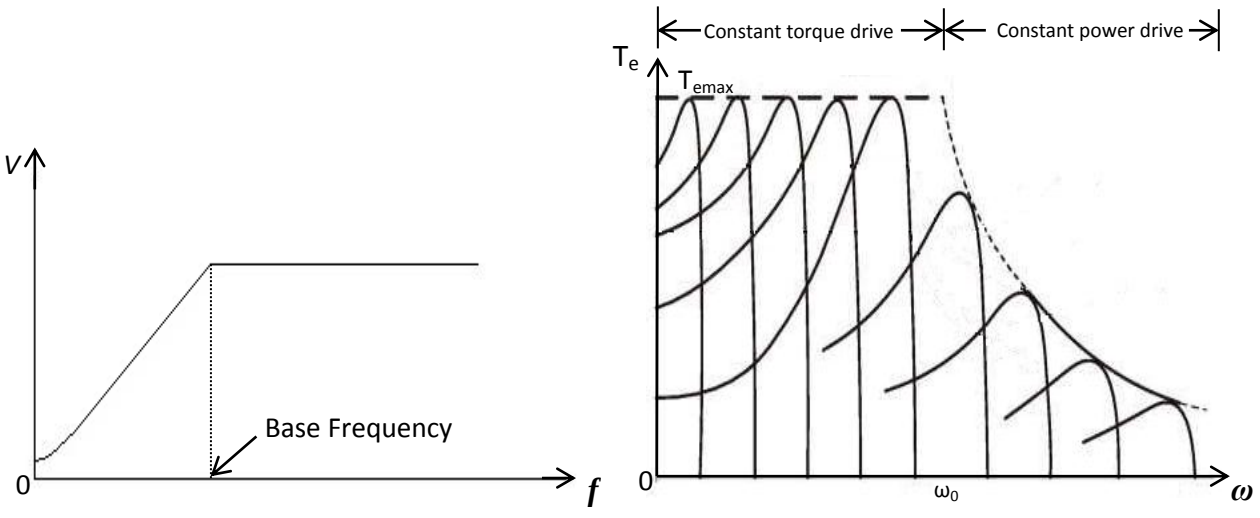
Terminal voltage $V_1 \propto \phi f$

It is evident that a reduction in the supply frequency without a change in the terminal voltage causes an increase in the air-gap flux (hence shifting the operating point of the motor towards saturation). The disadvantages associated with this increase are as follows:

- Significant increase in the magnetizing current.
- Distortion of line current and voltage.
- Increase in core loss and stator copper loss.
- Introduction of acoustic noise.

In order to avoid the above effects, the variable frequency control below the rated frequency is commonly carried out at rated air-gap flux by varying the terminal voltage with frequency in such a manner as to maintain (V/f) ratio constant at the rated value.

It is worthwhile to note that with a constant (V/f) ratio, motor develops a constant maximum torque. In an ideal motor, the stator magnetic circuit would be entirely inductive and fixing a constant V/f ratio would maintain a constant flux. But, the real motor has resistance in series with magnetizing inductance. For low speeds/frequencies, the stator resistance drop r_1 will become significant as compared to stator reactance drop x_1 and thus the maximum torque will have a lower value in this region. The same maximum torque can be retained by enhancing (V/f) ratio at low frequencies, which clearly justifies the following V-f relation curve having a non-zero V for zero f.



(a) *V-f* relation **(b)** Torque-Speed Characteristics

Fig 3: Variable Frequency Control

The reason for adopting the Volts/Hertz speed control above the other approaches can primarily be attributed to the following features:

(i) Speed can be varied from zero to above base speed, both during motoring and braking operations.
(ii) Decent dynamic response obtained as operation during transients can be carried out at the maximum torque with reduced current.
(iii) Clearly, the operation is restricted between synchronous speed and maximum torque point at all frequencies. That's why the copper losses are low and consequently, efficiency and power factor are high.

The VFVS (Variable frequency variable voltage source) for implementing this scheme can either be a VSI or cycloconverter. The latter employs a large number of thyristors and hence is cost-effective only when used in high power drives [11]. `

2.4 V/f Control Overview

The Volts/Hertz control is basically a scalar control technique where only the magnitudes of the control variables are varied (in this case, voltage and frequency). Scalar control, though easier to implement than vector control, provides inferior dynamic performance. The former cannot operate at peak performance under dynamically varying loads. In variable-speed applications in which a small variation of motor speed with loading is permissible, scalar control scheme can produce satisfactory performance. However, if precision control is required, then using vector control system is essential.

In scalar control, very little knowledge of the motor is required for frequency control. Consequently, this control is in wide use. Nevertheless, the loophole lies in the point that the torque developed is load dependent since it is not controlled directly. Besides, the transient response of such a control is sluggish due to the predefined switching configuration of the inverter.

From the torque-speed characteristic curves, it can be seen that the same torque at the same value of slip speed will be obtained if we operate at a constant air gap flux. This, in fact, is the basis for constant Volts/Hertz control of an induction motor. This type of control may be executed either in open loop or in closed loop.

Several real-life motor control applications do not demand a high dynamic performance, as long as the speed can be efficiently changed in the full range. This allows usage of a sinusoidal steady state model of the induction motor, in which the magnitude of the stator flux is proportional to the ratio between the magnitude and the frequency of the stator voltage. As long as this ratio is kept constant, the stator flux will remain constant, and as a result the motor torque will solely depend on the slip frequency.

2.5 Induction Motor Dynamic Model

The mathematical model of an induction machine can be obtained by first describing it as a coupled stator and rotor polyphase circuit in terms of so-called phase variables, viz. stator currents i_{as} , i_{bs} , and i_{cs} ; rotor currents i_{ar} , i_{br} , and i_{cr} ; the rotor speed ω_m; and the angular displacement θ between stator and rotor windings. The inductance matrix, which is a function of position θ, is a representative of the magnetic coupling. The matrix expression of the machine equations are formulated in Simulink language.

Next, the original stator and rotor *abc* frames of reference are transformed into a common *k* or *dq* frame in which the new variables for voltages, currents, and fluxes can be viewed as 2-D space vectors. In this common frame the inductances become constant independent of position. This is clear from Fig. 4 which illustrates various reference frames (coordinate systems): the triplet [A_s B_s C_s] designates a three-phase system attached to the stator while the pair [a_s b_s] links to an equivalent two-phase system. The possible selections of dq frames include:

a) Stator frame where $\omega_k = 0$
b) Rotor frame where $\omega_k = \omega_m$
c) Synchronous frame associated with frequency ω_s (conceivably time varying) of the stator excitation.
d) Rotor flux frame in which the d-axis lines up with the direction of the rotor flux vector.

The choice of the common *dq* frame is usually prescribed by the symmetry constraints levied by the construction and excitation of the machine. Owing to the complete symmetry encountered in a three – phase induction machine and balanced sinusoidal excitation, any one of the frames can be used, although the synchronous frame is more convenient as all signals appear as constant dc in steady state. In the presence of asymmetry, the common frame is attached to the asymmetrical member; in this case induction motor with unbalanced excitation or asymmetrical stator windings with be modeled in the stator frame. In the common *dq* frame, the machine dynamic equations come into view as differential equations with constant coefficients (i.e., independent of rotor position) and nonlinearities narrowed to products of variables associated with speed voltages and torque components.

ω denotes the rotational speed or angular frequency of a frame (in electrical rad/s) with respect to the stationary rotor. The angular position is obtained by integrating speed over time, that is

$$\theta = \int \omega \, dt \tag{2.3}$$

Initial conditions are established by specifying a steady-state operating condition. We will come across the simplest case in the simulation of the starting of a motor for which all initial conditions are zero. Mostly, a specified operating condition can only be obtained after running the simulation for a time that depends on the starting setting of the initial conditions. Normally, with power electronic input signals, a steady-state condition is reached when an output signal waveform is repeated every switching cycle so that the values at the beginning and end of a cycle

are equal. The so-called final state can be saved once a steady-state condition is attained and used later as the initial state in a reconditioned simulation which now comprises the specified time sequenced input events.

2.5.1 Space Vector Model of the Induction Machine (SI units)

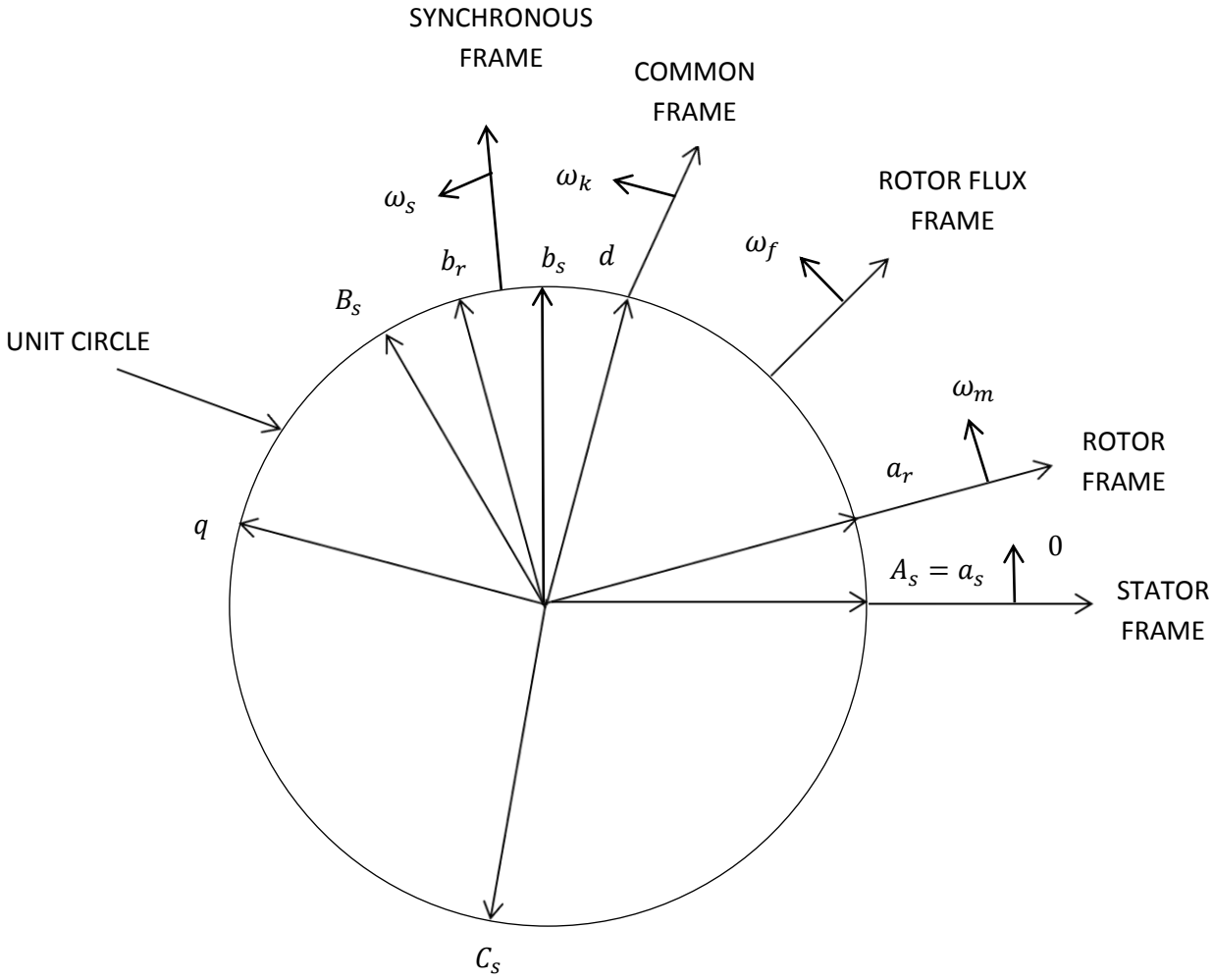


Fig 4: Reference frames in induction machine analysis

2.5.1.1 Electrical System Equations

$$\bar{v}_S = R_S \bar{\imath}_S + \frac{d\bar{\lambda}_S}{dt} + \omega_K M \bar{\lambda}_S \tag{2.4}$$

$$\bar{v}_r = R_r \bar{\imath}_r + \frac{d\bar{\lambda}_r}{dt} + (\omega_K - \omega_m) M \bar{\lambda}_r \tag{2.5}$$

where the space vector $\bar{f} = [f_d \quad f_q]^T$ and the $\frac{\pi}{2}$ rotational operator $M = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$

2.5.1.2 Flux Linkage-Current Relations

$$\bar{\lambda}_S = L_S\bar{\imath}_S + L_m\bar{\imath}_r \quad (2.6)$$

or $$\bar{\imath}_S = \Gamma_S\bar{\lambda}_S - \Gamma_m\bar{\lambda}_r \quad (2.7)$$

Also, $$\bar{\lambda}_r = L_m\bar{\imath}_S + L_r\bar{\imath}_r \quad (2.8)$$

or $$\bar{\imath}_r = -\Gamma_m\bar{\lambda}_S + \Gamma_r\bar{\lambda}_r \quad (2.9)$$

where $$L_S = L_m + L_{Sl} \quad (2.10)$$

$$L_r = L_m + L_{rl} \quad (2.11)$$

and

$$\Gamma_S = \frac{L_r}{\Delta} \quad (2.12)$$

$$\Gamma_r = \frac{L_S}{\Delta} \quad (2.13)$$

$$\Gamma_m = \frac{L_m}{\Delta} \quad (2.14)$$

where $$\Delta = L_mL_{Sl} + L_mL_{rl} + L_{Sl}L_{rl}$$

2.5.1.3 Mechanical System Equations

$$T_e = J\frac{d\omega_{mec}}{dt} + B_m\omega_{mec} + T_L \quad (2.15)$$

where $T_e = k(\bar{\lambda}_S \otimes \bar{\imath}_S) = k(M\bar{\lambda}_S \bullet \bar{\imath}_S) = k(\bar{\lambda}_{dS}\bar{\imath}_{qS} - \bar{\lambda}_{qS}\bar{\imath}_{dS})$ (2.16)

or $T_e = k(\bar{\imath}_r \otimes \bar{\lambda}_r) = kL_m(\bar{\imath}_r \otimes \bar{\imath}_S) = k\frac{L_m}{L_r}(\bar{\lambda}_r \otimes \bar{\imath}_S) = k\Gamma_m(\bar{\lambda}_r \otimes \bar{\lambda}_S)$ (2.17)

and $\omega_{mec} = \frac{2}{p}\omega_m$, $k = \frac{3}{2}\frac{p}{2}$

2.5.1.4 Nomenclature

Symbol	Meaning
$\bar{v}$	Voltage space vector [V]
$\bar{\imath}$	Current space vector [A]
$\bar{\lambda}$	Flux linkage space vector [Wb]
R	Resistance [Ω]

Symbol	Meaning
L	Inductance [H]
Γ	Inverse inductance [H^{-1}]
f_0	Base frequency [Hz]
$\omega_0 = 2\pi f_0$	Base frequency [rad/s]
ω_k	Speed of dq frame [rad/s]
ω_m	Rotor speed [rad/s]
T_e	Electromagnetic torque [N.m]
T_L	Load torque [N.m]
J	Moment of inertia [$kg.m^2$]
P	Number of poles

Operators: $\otimes \rightarrow$ Cross product $\bullet \rightarrow$ Dot product M $\rightarrow$ Rotation

Subscripts: s $\rightarrow$ Stator r $\rightarrow$ Rotor d $\rightarrow$ Direct axis q $\rightarrow$ Quadrature axis

2.5.1.5 Simulink Block Diagram Model

The aforementioned mathematical equations are shown in Fig 25 in a block diagram form that preserves the one-to-one conformity between the 2D space vectors of the equations and the vectored signals (of width 2) appearing in the Simulink depiction. The flux linkages are selected as state variables in the simulation. Apart from the scalar speed signals, all the variables are 2-element vectors.

2.6 Summary

This chapter throws light upon some of the basics of induction motor, which include its constructional details, working and in particular its pluses over conventional dc motors. It is a singly-fed motor unlike the synchronous motor which calls for ac supply on the stator side and dc excitation on the rotor. The torque developed in this motor originates from current induced in the rotor which is only feasible at non-synchronous speed; hence it is also known as *asynchronous* machine. In view of the fact that the air-gap excitation current is much larger in an induction motor than in a transformer for the same VA rating, it inherently has a power factor less than unity. Out of all the methods stated for speed-regulation of an induction motor, the

method of variable frequency control is the one largely preferred owing to the fact that it allows a variable speed drive with good running and transient performance to be obtained from a squirrel cage induction motor, which is economical, robust, reliable and long-lasting.

As far as the system dynamics are concerned, here a 2D space vector is represented by a column vector having 2 real elements such as $\bar{f} = [f_d \quad f_q]^T$. While the manipulation of equations using complex quantities is somewhat easier to perform, the process veils the underlying physical and geometric interpretation achievable with real vectors in a real 2D plane. Thereafter, for computer simulation, the differential equations are recast into real forms by assembling real and imaginary parts.

CHAPTER 3

Fuzzy Set Theory

"All natural languages are inherently ambiguous."

(Anonymous)

3.1 Fuzzy Logic as an Evolutionary Computational Tool

Fuzzy logic, first introduced by Lotfi A. Zadeh [3] in 1965, embodies human-like thinking into a control system. A fuzzy controller employs a mode of approximate reasoning resembling the decision making route of humans, that is, the process people use to infer conclusions from what they know. Fuzzy control has been primarily applied to the control of processes through fuzzy linguistic descriptions stipulated by membership functions.

The conventional Boolean logic has been extended to deal with the concept of partial truth - truth values which exist between "completely true" and "completely false", and what we shall be referring to as *fuzzy logic* [3]. This is achieved through the concept of degree of membership. The essence of fuzzy logic rests on a set of linguistic if-then rules, like a human operator. It has met a growing interest in many motor control applications due to its non-linearity handling features and independence of plant modeling. Moreover, the fuzzy logic concepts play a vital role in developing controllers for the plant since it isn't needy of the much complicated hardware and all it necessitates are only some set of rules.

3.2 Classical Set and Fuzzy Set: A Comparison

Let X be a space of objects (called universe of discourse or universal set) and x be a generic element of X.

A classical set A (A is a subset of X), is defined as a collection of elements or objects $x \in X$, such that each x can either belong or not belong to the set A. By defining a *characteristic function* for each element x in X, we can represent the classical set A by a set of ordered pairs $(x, 0)$ or $(x, 1)$ which indicates $x \notin A$ or $x \in A$, respectively.

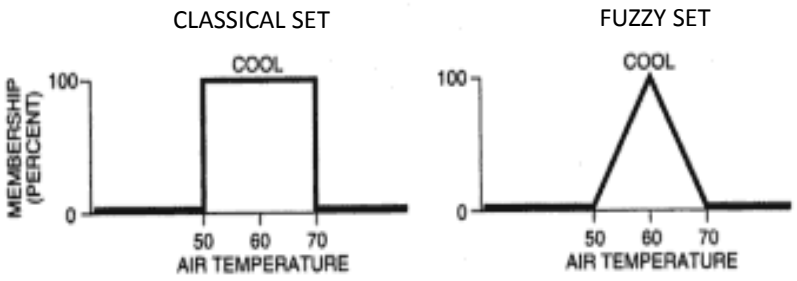


Fig 5: Example of Classical Set and Fuzzy set

In spite of being an important tool for the engineering sciences, classical sets fail to replicate the nature of human conceptions, which tend to be abstract and vague.

A fuzzy set [3] conveys the degree to which an element belongs to a set. In other words,
if X is a collection of objects denoted generically by x, then a fuzzy set A in X is defined as a set of ordered pairs:

$$A = \{(x, \mu_A(x))|x \in X\} \tag{3.1}$$

where $\mu_A(x)$ is known as the *membership function* for the fuzzy set A. MF serves the purpose of mapping each element of X to a membership grade (or membership value) between 0 and 1.Clearly, if the value of $\mu_A(x)$ is restricted to either 0 or 1, then A is reduced to a classical set and $\mu_A(x)$ is the characteristic function of A.

3.3: Fuzzy Sets with a Continuous Universe

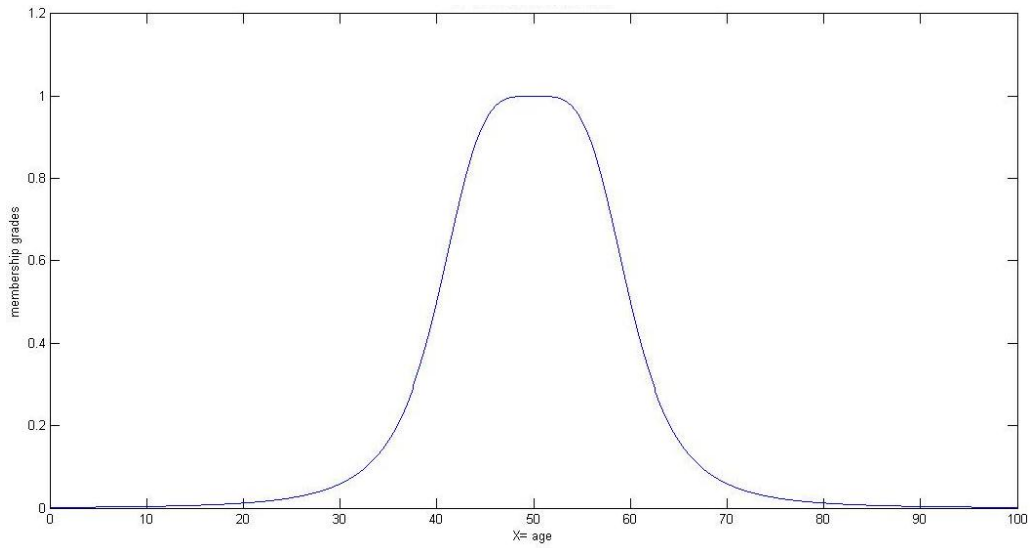


Fig 6: Membership Function on a Continuous Universe

Let X be the set of possible ages for human beings. Then the fuzzy set A = "about 50 years old" may be expressed as

$$A = \{(x, \mu_A(x))|x \in X\}$$

where,
$$\mu_A(x) = \frac{1}{1+(\frac{x-50}{10})^4} \tag{3.2}$$

The aforementioned example clearly expresses the dependence of the construction of a fuzzy set on two things:

- Identifying a suitable universe of discourse.
- Laying down a suitable membership function.

At this point, it is imperative to state that the specification of membership functions is *subjective*, meaning that membership functions stated for the same notion by different persons will tend to vary noticeably. **Subjectivity** and **non-randomness** differentiate the study of fuzzy sets from probability theory. Latter deals with tangible handling of random phenomena.

***Crisp variable*:** A crisp variable is a physical variable that can be measured through instruments and can be assigned a crisp or discrete value, such as a temperature of 30 ^{0}C, an output voltage of 8.55 V etc.

***Linguistic variable*:** When the universe of discourse is a continuous space, the common practice is to partition X into several fuzzy sets whose MFs cover X in a more or less uniform manner. These fuzzy sets, which usually carry names that conform to adjectives appearing in our daily linguistic usage, such as "large", "medium" or "small", are called linguistic values. Consequently, the universe of discourse X is often called the linguistic variable.

3.4 Fuzzy Set-Theoretic Operations

The most elementary operations on classical sets include union, intersection and complement. Analogous to these operations, fuzzy sets also have similar operations [3] which are explained below.

3.4.1 Containment or Subset

Fuzzy set A is contained in fuzzy set B (or, equivalently, A is a subset of B) iff $\mu_A(x) \leq \mu_B(x)$ for all x. The following figure clarifies this concept.

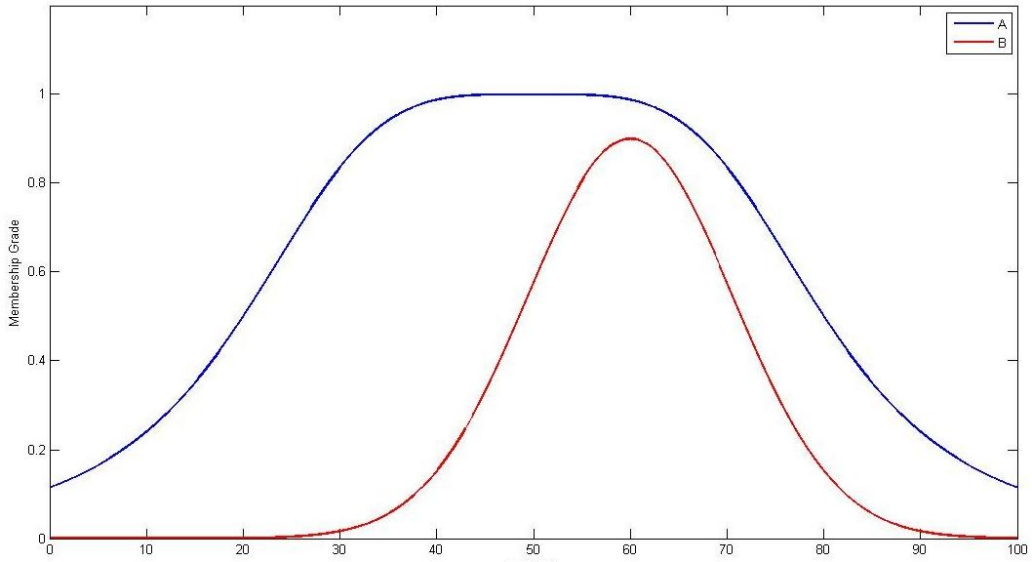


Fig 7: The concept of containment or subset

3.4.2 Union (Disjunction)

The **union** of two fuzzy sets A and B is a fuzzy set C, written as C = A ∪ B or C = A OR B, whose MF is related to those of A and B by

$$\mu_C(x) = \max\left(\mu_A(x), \mu_B(x)\right) = \mu_A(x) \vee \mu_B(x) \tag{3.3}$$

Equivalently, union is the *smallest* fuzzy set containing both A and B. Then again, if D is any fuzzy set encompassing both A and B, then it also contains A ∪ B. A union of two fuzzy sets A and B is shown in Fig 7 (b).

3.4.3 Intersection (Conjunction)

The **intersection** of two fuzzy sets A and B is a fuzzy set C, written as C = A ∩ B or C = A AND B, whose MF is related to those of A and B by

$$\mu_C(x) = \min\left(\mu_A(x), \mu_B(x)\right) = \mu_A(x) \wedge \mu_B(x) \tag{3.4}$$

Analogous to the definition of union, intersection of A and B is the *largest* fuzzy set which is contained in both A and B. An intersection of two fuzzy sets A and B is shown in Fig 7 (c).

3.4.4 Complement (Negation)

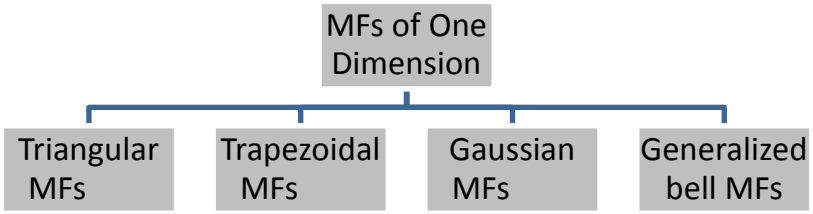
The **complement** of fuzzy set A, designated by $\bar{A}$ (¬A, NOT A), is defined as

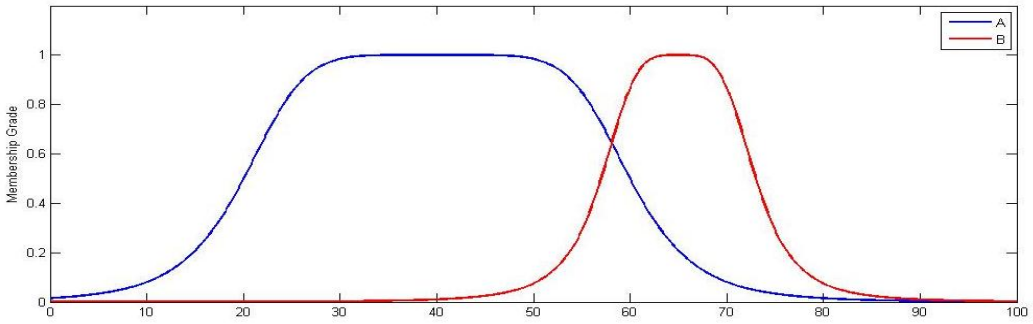
$$\mu_{\bar{A}}(x) = 1 - \mu_A(x) \tag{3.5}$$

3.5 Formulating Membership Functions

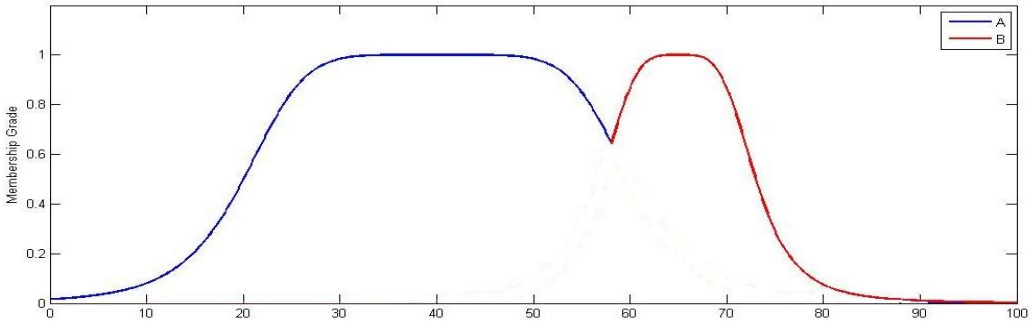
Any membership function completely characterizes the fuzzy set that it belongs to. A convenient and succinct way to define an MF is to express it as a mathematical function. In order to define fuzzy membership function, designers choose many different shapes based on their preference and know-how.

Different classes of parameterized membership functions [14] commonly used are:

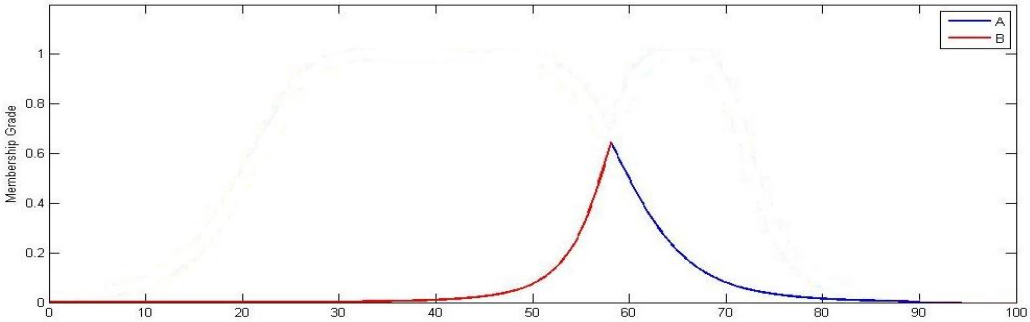




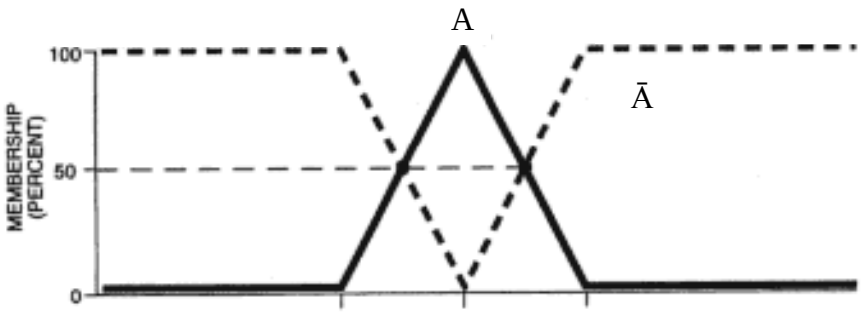
(a) Two Fuzzy sets A and B



(b) A ∪ B



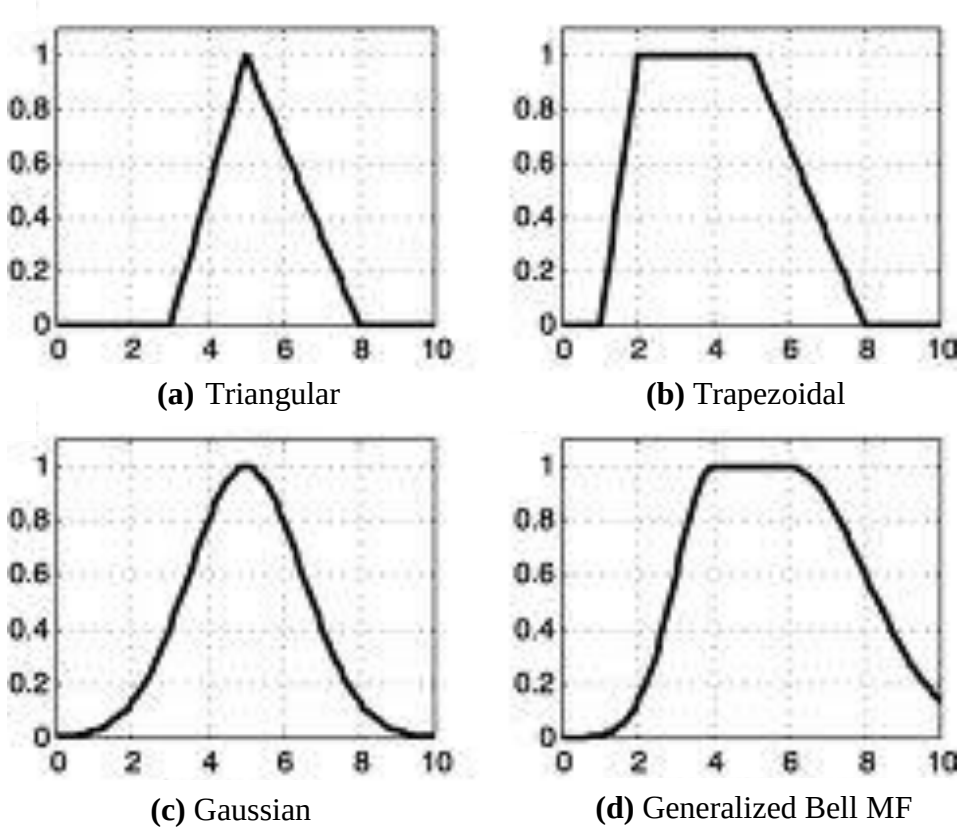
(c) A ∩ B



(d) Fuzzy set A and its Complement Ā

Fig 8: Operations on Fuzzy sets

Among the alternatives just mentioned, the most popularly used MFs in real-time implementations are triangular and trapezoidal because of the fact that these are easy to represent the designer's idea and require low computation time.



(a) Triangular **(b)** Trapezoidal

(c) Gaussian **(d)** Generalized Bell MF

Fig 9: Examples of four classes of parameterized MFs

The aforementioned classes of parameterized MFs of one dimension are defined for MFs with a single input. MFs of higher dimensions can be defined similarly as per the increase in the number of inputs.

3.6 Summary

This chapter defines the necessity of fuzzy logic, introduces fuzzy sets and corresponding set operations (AND, OR, and NOT), as well as describes membership function representations and their types.

A fuzzy set is a set without a crisp periphery. That is, the switch from "belong to a set" to "not belong to a set" is steady, and this smooth transition is characterized by membership functions that give fuzzy sets flexibility in modeling universally used linguistic expressions. These sets [14] play a significant role in human thinking, particularly in the domains of pattern recognition, communication of information and perception. Fuzziness does not come from the randomness of

the constituent members of the sets, but from the uncertain and imprecise nature of abstract thoughts and concepts. Fuzzy set is simply an extension of a classical set in which the characteristic function is allowed to have values between 0 and 1, which denotes the degree of membership of an element in a given set. The specification of membership functions is subjective, which comes from individual differences in perceiving nonconcrete models. The *universe of discourse* may consist of discrete objects or continuous space, which is totally covered by the MFs and the transition from one MF to another, is smooth and gradual. The union, intersection and negation operations perform exactly as that for crisp sets if the values of the membership functions are restricted to either 0 or 1.

CHAPTER 4

Fuzzy Logic Controller

""Research is formalized curiosity"

(Zora Neale Hurston)

4.1 Introduction

One of the reasons for the popularity of Fuzzy Logic Controllers is its logical resemblance to a human operator. It operates on the foundations of a knowledge base which in turn rely upon the various if then rules, similar to a human operator [6]. Unlike other control strategies, this is simpler as there is no complex mathematical knowledge required. The FLC requires only a qualitative knowledge of the system thereby making the controller not only easy to use, but also easy to design.

4.2 Application Areas of Fuzzy Logic Controllers

The fuzzy logic Controllers are basically put to use when [5]:

1) The system is highly non-linear thereby making the making the mathematical modeling of the system very arduous.
2) The analytical form of the system is not provided, instead a linguistic form is provided.
3) The precise identification of the system parameters.
4) The system behavior has a vague characteristic under precisely defined conditions. [5]
5) The conditions themselves are vague.

4.3 Components of FLC

The inputs to a Fuzzy Logic Controller are the processed with the help of linguistic variables which in turn are defined with the aid of membership functions. The membership functions are chosen in such a manner that they cover the whole of the universe of discourse. To avoid any discontinuity with respect to minor changes in the inputs, the adjacent fuzzy sets must overlap each other [7]. Because of a small time constant in Fuzzy Logic Controllers, this criterion is very important in the design of the same.

There are basically three essential segments in Fuzzy Logic Controller viz.

1. Fuzzification block or Fuzzifier.
2. Inference System.
3. Defuzzification block or Defuzzufier.

4.3.1 Fuzzification Block or Fuzzifier

The first step towards designing a Fuzzy Logic Controller is choosing appropriate inputs which will be fed to the same. These input variables should be such that, they represent the dynamical system completely. Then the function of the Fuzzifier comes into picture. As discussed before, instead of using numerical variables, fuzzy logic uses linguistic variables for processing information. But since the inputs to the FLC are in the form of numerical variables (or in other words, crisp sets), they need to be converted into linguistic variables. This function of converting these crisp sets into fuzzy sets (linguistic variables) is performed by the Fuzzifier.

The fuzzification technique involves outlining the membership functions for the inputs. These membership functions should cover the whole universe of discourse and each one represents a fuzzy set or a linguistic variable. The crisp inputs are thus transformed into fuzzy sets. Triangular MF, Trapezoidal MF, Bell MF, Generalized Bell MF or Sigmoidal MF [14] can be used. Even a hybrid of any of the above Membership Functions can be used for fuzzification.

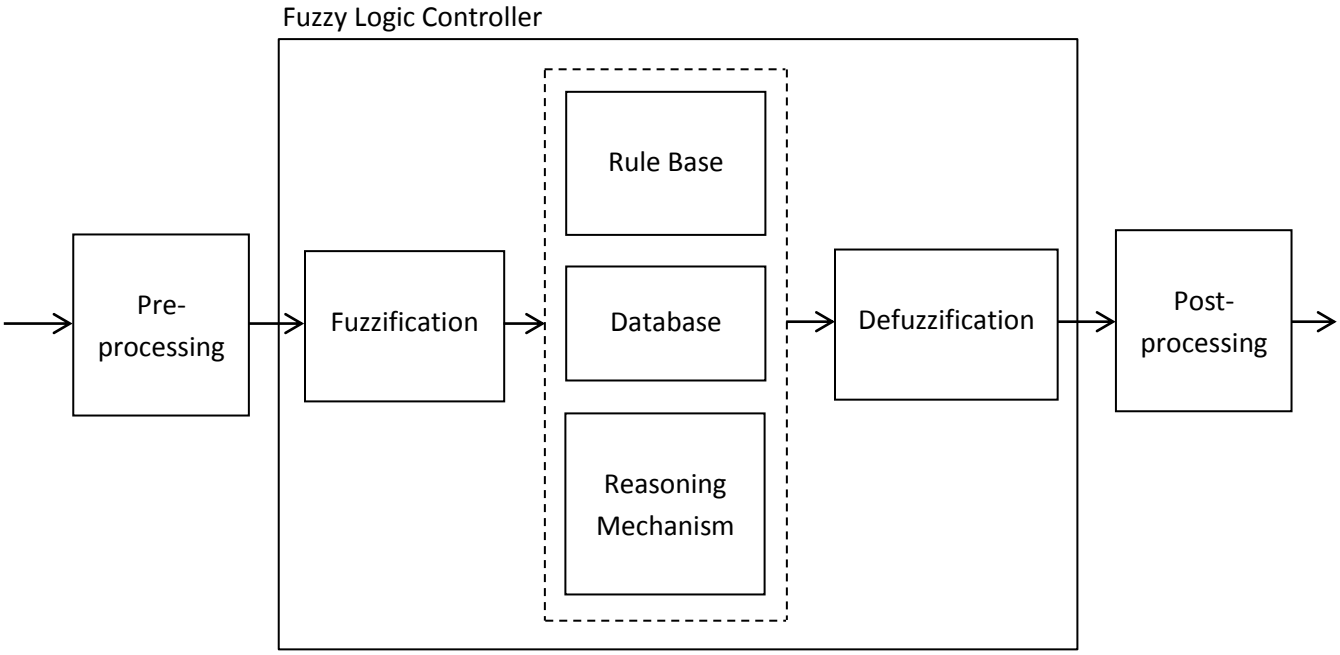


Fig 10: Fuzzy Logic Controller Structure

4.3.2 Inference System

The inference system of a Fuzzy Logic Controller consists of the following three paradigms [14]:

1. **Rule Base**: - It consists of a number of If-Then rules. The If side of the rule is called the antecedent and the Then side is called the consequence. These rules are very much similar to the Human thought process and the computer uses the linguistic variables, derived after fuzzification for execution of the rules. They very simple to understand and write and hence the programming for the fuzzy logic controller becomes very simple. The control strategy is stored in more or less the normal language.

2. **Database**: - It consists of the all the defined membership functions that are to be used by the rules.

3. **Reasoning Mechanism**: - It performs the inference procedure on the rules and the data given to provide a reasonable output. It is basically the codes of the software which are process the rules and the all the knowledge based on a particular situation. It exercises a human brain type of attribute to methodically carry out the inference steps for processing the information.

4.3.3 Defuzzification Block or Defuzzifier

A defuzzifier performs the exact opposite function of a fuzzifier. It transforms the fuzzy variables (which are obtained as output after processing of the inputs) to crisp sets. The defuzzifier is necessary because in the real world the crisp values can only be taken as inputs to the other systems. Even though the fuzzy sets resemble the human thought process, their functionality is limited only to the above processes.

A defuzzifier is generally required only when the Mamdani Fuzzy Model is used for designing a controller. There are other types of architectures that can be used are:

- Tagaki-Sugeno Fuzzy Model [14].
- Tsukamoto Fuzzy Model [14].

Mamdani model is preferred here because it follows the Compositional Rule of Inference [14] strictly in its fuzzy reasoning mechanism. Unlike the Mamdani model, the outputs are defined with the help of a specific function for the other two models (first order polynomial in the input variables) and hence the output is crisp instead of fuzzy. This is counterintuitive since a fuzzy model should be able to propagate the fuzziness from inputs to outputs in an appropriate manner [14].

There are five basic defuzzification strategies and they are defined as follows:

1. **Centroid of Area (COA):**
   It is one of the most popular techniques used for defuzzification, as it is reminiscent of the calculation of expected values of probability distributions. It can be defined as follows:

$$z_{COA} = \frac{\int_Z \mu_A(z)\, z \; dz}{\int_Z \mu_A(z) \; dz} \tag{4.1}$$

   where $\mu_A(z)$ is the aggregated output MF.

2. **Bisector of Area (BOA):**
   It satisfies the equation:

$$\int_{\alpha}^{z_{BOA}} \mu_A(z)\, dz \;=\; \int_{z_{BOA}}^{\beta} \mu_A(z)\, dz \qquad (4.2)$$

Where $\alpha = min\{z|z \in Z\}$ and $\beta = max\{z|z \in Z\}$. That is the vertical line z = z_{BOA} partitions the region between z = α, z = β, y = 0 and y = $\mu_A(z)$ into two regions with same area.

3. **Mean of Maximum (MOM):**
   z_{MOM} is the average of the maximizing z at which the MF reaches a maximum μ^*. It can be represented as follows:

$$z_{MOM} = \frac{\int_{z'} z\ dz}{\int_{z'} dz} \qquad (4.3)$$

   where z' = {z | μ_A (z) = μ^*}

4. **Smallest of Minimum (SOM):**
   z_{SOM} is the minimum of the maximizing z (in terms of magnitude).

5. **Largest of Maximum (LOM):**
   z_{LOM} is the maximum of the maximizing z (in terms of magnitude).

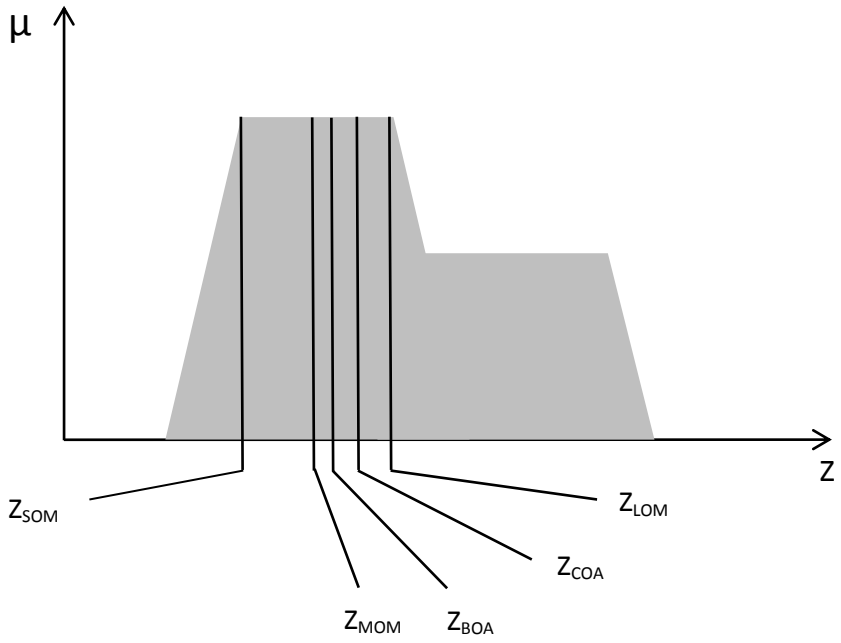


Fig 11: Various Defuzzification schemes for obtaining crisp outputs

The various deffuzication techniques have been explained with the help of the figure shown above. The last two defuzzification techniques are rarely used because of their biased nature. The most wide used technique is the Centre of Area (COA) method [14].

4.4 Summary

Thus, we have seen that the designing of a Fuzzy Logic Controller (using the Mamdani Fuzzy Model) requires:

1. The selection of appropriate inputs and their fuzzification.
2. The definition of the input and output membership functions.
3. The definition of the Fuzzy Rule Base.
4. The defuzzification of the output obtained after the processing of the linguistic variables with the help of a proper defuzzification technique.

Each of them has to be designed based on the result that is desired from the system.

Chapter 5

Fuzzy Logic Controller Design

"The ability to simplify means to eliminate the unnecessary so that the necessary may speak"

(Hans Hoffmann)

5.1 Block Diagram for Speed Control of Induction Motor

The Block diagram employing speed control of an induction motor is shown in Fig 12. It can be seen that scalar method is employed in the given block diagram. The frequency and supply voltage of the induction motor are varied such that it operates at steady state, at the desired speed. In the scalar control method, both the input and output commands are speed, unlike the Vector control mode where it is torque/flux and reference current, respectively. Even though vector control drives provide excellent performance in terms of dynamic speed regulation, implementation of the same is tedious owing to on-line coordinate transformations that convert line currents into two axis representation and vice versa [8]. Hence, the use of scalar control method is used in this case.

As shown in the block diagram, ω_m^* is chosen as the reference signal. The use of speed as reference signal is justified as the output of the system is speed and our aim is to control the speed of the induction motor. A tacho-generator, attached to the shaft of the induction motor, provides the current speed of the motor (ω_m) which is compared with the reference speed (ω_m^*), thus providing us with the speed error (**e**). This mechanism is called the feedback mechanism [15]. A feedback mechanism is used to provide the quality of automation to the control system. The information about the instantaneous state of the output is fed back to the input which in turn is used to revise the same in order to achieve a desired output.

Change-of-error (Δ**e**), that is, the derivative of speed error is computed and both **e** and Δ**e** are fed to the fuzzifier for fuzzification. The inference system then processes these two fuzzy inputs using the fuzzy control rules and the database, which are defined by the programmer based on the chosen membership function and fuzzy rule table, to give an output fuzzy variable. The fuzzy output thus obtained is defuzzified by the defuzzifier to give a crisp value, i.e. change-of-control (ω_{sl}). This ω_{sl} is then added to the motor speed (ω_m) which in turn forms the input ω_e^* to the Voltage Source Inverter and V/f controller.

$$\omega_e^* = \omega_{sl} + \omega_m \qquad (5.1)$$

The VSI (Voltage Source Inverter) receives voltage signal (V_s^*) from the V/f controller and ω_e^* (frequency signal). The Voltage Source Inverter uses these inputs to generate a three phase voltage whose frequency and amplitude can be varied by the Fuzzy Logic Controller itself via the above mentioned process. The three phase voltage is fed to the induction motor which then runs with a speed which tends to follow the desired speed (reference speed, ω_m^*).

Before running the simulation in MATLAB/SIMULINK®, the Fuzzy Logic Controller is to be designed. This is done using the FIS editor. The membership functions and the rules have to be designed by the programmer so as to achieve the desired results. The FIS program thus generated is to be fed to the FLC before proceeding with the simulation.

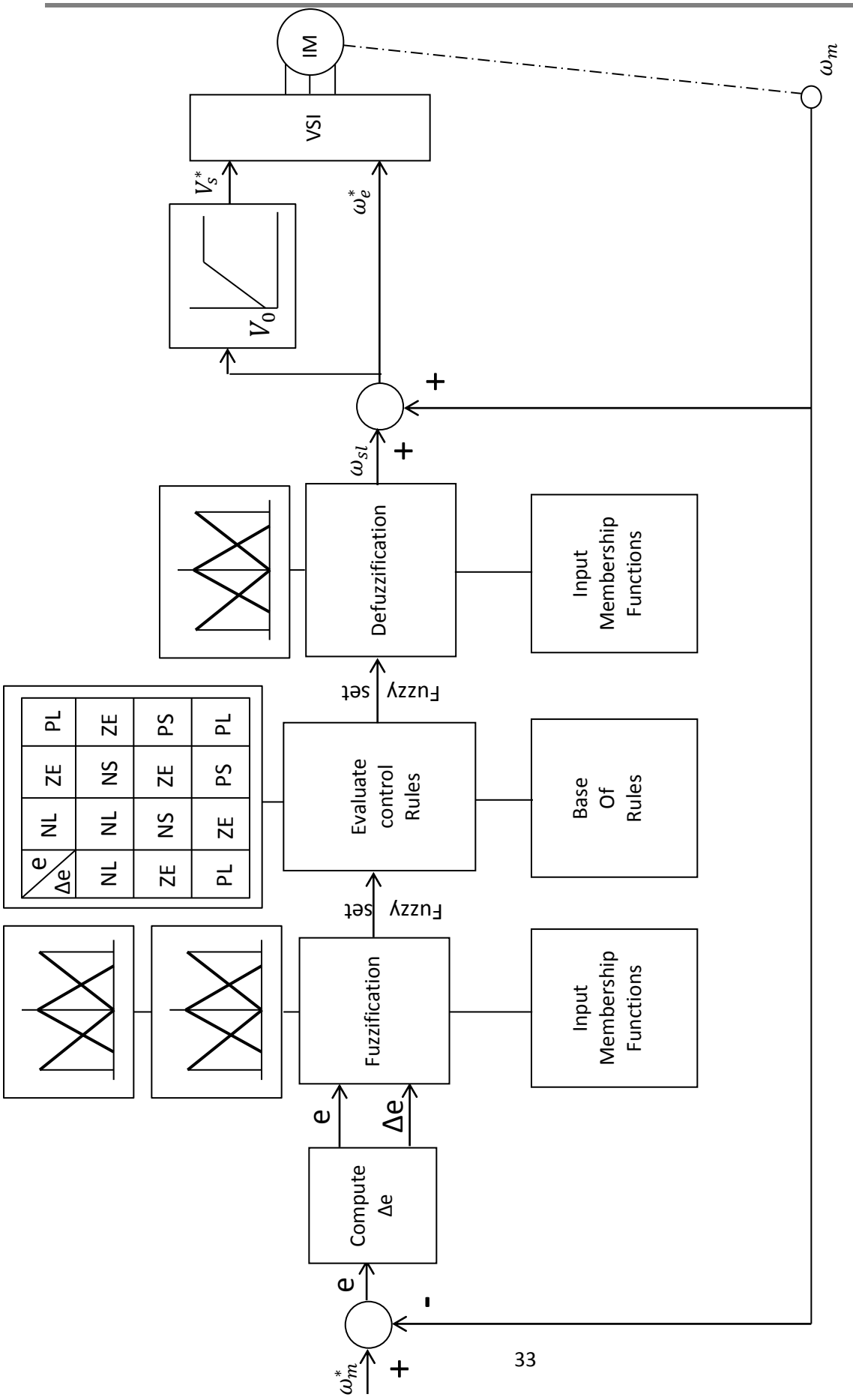


Fig 12: Block Diagram of the scalar IM control with the FLC architecture

5.2 Fuzzy Logic Controller Design

The design of a Fuzzy Logic Controller requires the choice of Membership Functions. The membership functions should be chosen such that they cover the whole universe of discourse. It should be taken care that the membership functions overlap each other. This is done in order to avoid any kind of discontinuity with respect to the minor changes in the inputs. To achieve finer control, the membership functions near the zero region should be made narrow. Wider membership functions away from the zero region provides faster response to the system. Hence, the membership functions should be adjusted accordingly [9]. After the appropriate membership functions are chosen, a rule base should be created. It consists of a number of Fuzzy If-Then rules that completely define the behaviour of the system. These rules very much resemble the human thought process, thereby providing artificial intelligence to the system.

5.3 Membership Function Design

5.3.1 Input Linguistic Variables

The inputs to the Fuzzy Logic Controller are:

1) Speed Error (e).
2) Change in Error (Δe) or derivative of speed error.

5.3.1.1 Fuzzy Sets and MFs for Input Variable Speed Error (e)

Table 1: Fuzzy sets and the respective membership functions for speed error (e)

Fuzzy set or label	Set Description	Range	Membersihp Function
NL (Negative Large)	Speed error is high in the negative direction.	-1.0 to -1.0 -1.0 to -0.8 -0.8 to -0.5	Trapezoidal
NM (Negative Medium)	Speed error is medium in the negative direction.	-0.8 to -0.5 -0.5 to -0.2	Triangular
NS (Negative Small)	Speed error is small in the negative direction.	-0.5 to -0.2 -0.2 to 0	Triangular
ZE (Zero)	Speed error is around zero.	-0.2 to 0 0 to 0.2	Triangular
PS (Positive Small)	Speed error is small in the positive direction.	0 to 0.2 0.2 to 0.5	Triangular
PM (Positive Medium)	Speed error is medium in the positive direction.	0.2 to 0.5 0.5 to 0.8	Triangular
PL (Positive Large)	Speed error is high in the positive direction.	0.5 to 0.8 0.8 to 1.0 1.0 to 1.0	Trapezoidal

5.3.1.2 Fuzzy Sets and MFs for Input Variable Change in Error (Δe)

Table 2: Fuzzy sets and the respective membership functions for Change in Error (Δe)

Fuzzy set or label	Set Description	Range	Membersihp Function
NL (Negative Large)	Speed error is high in the negative direction.	-1.0 to -1.0 -1.0 to -0.8 -0.8 to -0.5	Trapezoidal
NM (Negative Medium)	Speed error is medium in the negative direction.	-0.8 to -0.5 -0.5 to -0.2	Triangular
NS (Negative Small)	Speed error is small in the negative direction.	-0.5 to -0.2 -0.2 to 0	Triangular
ZE (Zero)	Speed error is around zero.	-0.2 to 0 0 to 0.2	Triangular
PS (Positive Small)	Speed error is small in the positive direction.	0 to 0.2 0.2 to 0.5	Triangular
PM (Positive Medium)	Speed error is medium in the positive direction.	0.2 to 0.5 0.5 to 0.8	Triangular
PL (Positive Large)	Speed error is high in the positive direction.	0.5 to 0.8 0.8 to 1.0 1.0 to 1.0	Trapezoidal

5.3.2 Output Linguistic Variable

These inputs are processed to obtain an output known as the Change of Control (ω_{sl}).

5.3.2.1 Fuzzy Sets and MFs for Output Variable Change of Control (ω_{sl})

Table 3: Fuzzy sets and the respective membership functions for Change of Control (ω_{sl})

Fuzzy set or Label	Range	Membership Function
NL (Negative Large)	-1.0 to -1.0 -1.0 to -0.8	Triangular
NLM (Negative Large Medium)	-1.0 to -0.8 -0.8 to -0.6	Triangular
NM (Negative Medium)	-0.8 to -0.6 -0.6 to -0.4	Triangular
NMS (Negative Medium Small)	-0.6 to -0.4 -0.4 to -0.2	Triangular
NS (Negative Small)	-0.4 to -0.2 -0.2 to 0	Triangular
ZE (Zero)	-0.2 to 0 0 to 0.2	Triangular

PS (Positive Small)	0 to 0.2 0.2 to 0.4	Triangular
PMS (Positive Medium Small)	0.2 to 0.4 0.4 to 0.6	Triangular
PM (Positive Medium)	0.4 to 0.6 0.6 to 0.8	Triangular
PLM (Positive Large Medium)	0.6 to 0.8 0.8 to 1.0	Triangular
PL (Positive Large)	0.8 to 1.0 1.0 to 1.0	Triangular

5.4 Rule Base Design for the Output (ω_{sl})

The Rule Base for deciding the output of the inference system consists of 49 If-Then rules in this case since there are 7 fuzzy sets in each of the inputs. The table representing the rule base is as follows:

Table 4: Fuzzy Rule Table for Output (ω_{sl})

Δe \ e	**NL**	**NM**	**NS**	**ZE**	**PS**	**PM**	**PL**
NL	NL	NL	NLM	NM	NMS	NS	ZE
NM	NL	NLM	NM	NMS	NS	ZE	PS
NS	NLM	NM	NMS	NS	ZE	PS	PMS
ZE	NM	NMS	NS	ZE	PS	PMS	PM
PS	NMS	NS	ZE	PS	PMS	PM	PLM
PM	NS	ZE	PS	PMS	PM	PLM	PL
PL	ZE	PS	PMS	PM	PLM	PL	PL

5.5 Design of the Fuzzy Logic Controller using MATLAB

While simulating the block diagram in MATLAB/SIMULINK®, the Fuzzy Logic Controller has to be programmed according to the aforementioned rules and knowledge base. The program is saved as an FIS file and it is later embedded into the Fuzzy Logic Controller. This FIS program can be checked with the help of FIS editor in MATLAB itself. The steps for the following are shown below, along with the membership functions, the rules and the surface plot viewed with the help of the FIS editor.

Step 1:

- The program for designing of the Fuzzy Logic Controller is written in a word file.
- The definitions for all the shown membership functions are written in the program.
- The 49 rules shown in tabular form in section 5.4 are written in the program according to the syntax provided by MATLAB.
- The document is saved with the extension **.fis**.

All the 49 If- Then Rules of the Rule Base used for the design of the Fuzzy Logic Controller are as follows:

IF (Error IS NL) AND (ChangeInError IS NL) THEN (ChangeOfControl IS NL)
IF (Error IS NM) AND (ChangeInError IS NL) THEN (ChangeOfControl IS NL)
IF (Error IS NS) AND (ChangeInError IS NL) THEN (ChangeOfControl IS NLM)
IF (Error IS ZE) AND (ChangeInError IS NL) THEN (ChangeOfControl IS NM)
IF (Error IS PS) AND (ChangeInError IS NL) THEN (ChangeOfControl IS NMS)
IF (Error IS PM) AND (ChangeInError IS NL) THEN (ChangeOfControl IS NS)
IF (Error IS PL) AND (ChangeInError IS NL) THEN (ChangeOfControl IS ZE)
IF (Error IS NL) AND (ChangeInError IS NM) THEN (ChangeOfControl IS NL)
IF (Error IS NM) AND (ChangeInError IS NM) THEN (ChangeOfControl IS NLM)
IF (Error IS NS) AND (ChangeInError IS NM) THEN (ChangeOfControl IS NM)
IF (Error IS ZE) AND (ChangeInError IS NM) THEN (ChangeOfControl IS NMS)
IF (Error IS PS) AND (ChangeInError IS NM) THEN (ChangeOfControl IS NS)
IF (Error IS PM) AND (ChangeInError IS NM) THEN (ChangeOfControl IS ZE)
IF (Error IS PL) AND (ChangeInError IS NM) THEN (ChangeOfControl IS PS)
IF (Error IS NL) AND (ChangeInError IS NS) THEN (ChangeOfControl IS NLM)
IF (Error IS NM) AND (ChangeInError IS NS) THEN (ChangeOfControl IS NM)
IF (Error IS NS) AND (ChangeInError IS NS) THEN (ChangeOfControl IS NMS)
IF (Error IS ZE) AND (ChangeInError IS NS) THEN (ChangeOfControl IS NS)
IF (Error IS PS) AND (ChangeInError IS NS) THEN (ChangeOfControl IS ZE)
IF (Error IS PM) AND (ChangeInError IS NS) THEN (ChangeOfControl IS PS)
IF (Error IS PL) AND (ChangeInError IS NS) THEN (ChangeOfControl IS PMS)
IF (Error IS NL) AND (ChangeInError IS ZE) THEN (ChangeOfControl IS NM)
IF (Error IS NM) AND (ChangeInError IS ZE) THEN (ChangeOfControl IS NMS)
IF (Error IS NS) AND (ChangeInError IS ZE) THEN (ChangeOfControl IS NS)
IF (Error IS ZE) AND (ChangeInError IS ZE) THEN (ChangeOfControl IS ZE)
IF (Error IS PS) AND (ChangeInError IS ZE) THEN (ChangeOfControl IS PS)
IF (Error IS PM) AND (ChangeInError IS ZE) THEN (ChangeOfControl IS PMS)
IF (Error IS PL) AND (ChangeInError IS ZE) THEN (ChangeOfControl IS PM)

IF (Error IS NL) AND (ChangeInError IS PS) THEN (ChangeOfControl IS NMS)
IF (Error IS NM) AND (ChangeInError IS PS) THEN (ChangeOfControl IS NS)
IF (Error IS NS) AND (ChangeInError IS PS) THEN (ChangeOfControl IS ZE)
IF (Error IS ZE) AND (ChangeInError IS PS) THEN (ChangeOfControl IS PS)
IF (Error IS PS) AND (ChangeInError IS PS) THEN (ChangeOfControl IS PMS)
IF (Error IS PM) AND (ChangeInError IS PS) THEN (ChangeOfControl IS PM)
IF (Error IS PL) AND (ChangeInError IS PS) THEN (ChangeOfControl IS PLM)
IF (Error IS NL) AND (ChangeInError IS PM) THEN (ChangeOfControl IS NS)
IF (Error IS NM) AND (ChangeInError IS PM) THEN (ChangeOfControl IS ZE)
IF (Error IS NS) AND (ChangeInError IS PM) THEN (ChangeOfControl IS PS)
IF (Error IS ZE) AND (ChangeInError IS PM) THEN (ChangeOfControl IS PMS)
IF (Error IS PS) AND (ChangeInError IS PM) THEN (ChangeOfControl IS PM)
IF (Error IS PM) AND (ChangeInError IS PM) THEN (ChangeOfControl IS PLM)
IF (Error IS PL) AND (ChangeInError IS PM) THEN (ChangeOfControl IS PL)
IF (Error IS NL) AND (ChangeInError IS PL) THEN (ChangeOfControl IS ZE)
IF (Error IS NM) AND (ChangeInError IS PL) THEN (ChangeOfControl IS PS)
IF (Error IS NS) AND (ChangeInError IS PL) THEN (ChangeOfControl IS PMS)
IF (Error IS ZE) AND (ChangeInError IS PL) THEN (ChangeOfControl IS PM)
IF (Error IS PS) AND (ChangeInError IS PL) THEN (ChangeOfControl IS PLM)
IF (Error IS PM) AND (ChangeInError IS PL) THEN (ChangeOfControl IS PL)
IF (Error IS PL) AND (ChangeInError IS PL) THEN (ChangeOfControl IS PL)

The program for designing the Fuzzy Logic Controller using the FIS editor in MATLAB/SIMULINK® is as follows:

```
[System]
Name='rules'
Type='mamdani'
Version=2.0
NumInputs=2
NumOutputs=1
NumRules=49
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'

[Input1]
Name='Error'
```

```
Range=[-1 1]
NumMFs=7
MF1='NL':'trapmf',[-1 -1 -0.8 -0.5]
MF2='NM':'trimf',[-0.8 -0.5 -0.2]
MF3='NS':'trimf',[-0.5 -0.2 0]
MF4='ZE':'trimf',[-0.2 0 0.2]
MF5='PS':'trimf',[0 0.2 0.5]
MF6='PM':'trimf',[0.2 0.5 0.8]
MF7='PL':'trapmf',[0.5 0.8 1 1]

[Input2]
Name='ChangeInError'
Range=[-1 1]
NumMFs=7
MF1='NL':'trapmf',[-1 -1 -0.8 -0.5]
MF2='NM':'trimf',[-0.8 -0.5 -0.2]
MF3='NS':'trimf',[-0.5 -0.2 0]
MF4='ZE':'trimf',[-0.2 0 0.2]
MF5='PS':'trimf',[0 0.2 0.5]
MF6='PM':'trimf',[0.2 0.5 0.8]
MF7='PL':'trapmf',[0.5 0.8 1 1]

[Output1]
Name='ChangeOfControl'
Range=[-1 1]
NumMFs=11
MF1='NL':'trimf',[-1 -1 -0.8]
MF2='NLM':'trimf',[-1 -0.8 -0.6]
MF3='NM':'trimf',[-0.8 -0.6 -0.4]
MF4='NMS':'trimf',[-0.6 -0.4 -0.2]
MF5='NS':'trimf',[-0.4 -0.2 0]
MF6='ZE':'trimf',[-0.2 0 0.2]
MF7='PS':'trimf',[0 0.2 0.4]
MF8='PSM':'trimf',[0.2 0.4 0.6]
MF9='PM':'trimf',[0.4 0.6 0.8]
MF10='PML':'trimf',[0.6 0.8 1]
MF11='PL':'trimf',[0.8 1 1]

[Rules]
1 1, 1 (1) : 1
```

```
2 1, 1 (1) : 1
3 1, 2 (1) : 1
4 1, 3 (1) : 1
5 1, 4 (1) : 1
6 1, 5 (1) : 1
7 1, 6 (1) : 1
1 2, 1 (1) : 1
2 2, 2 (1) : 1
3 2, 3 (1) : 1
4 2, 4 (1) : 1
5 2, 5 (1) : 1
6 2, 6 (1) : 1
7 2, 7 (1) : 1
1 3, 2 (1) : 1
2 3, 3 (1) : 1
3 3, 4 (1) : 1
4 3, 5 (1) : 1
5 3, 6 (1) : 1
6 3, 7 (1) : 1
7 3, 8 (1) : 1
1 4, 3 (1) : 1
2 4, 4 (1) : 1
3 4, 5 (1) : 1
4 4, 6 (1) : 1
5 4, 7 (1) : 1
6 4, 8 (1) : 1
7 4, 9 (1) : 1
1 5, 4 (1) : 1
2 5, 5 (1) : 1
3 5, 6 (1) : 1
4 5, 7 (1) : 1
5 5, 8 (1) : 1
6 5, 9 (1) : 1
7 5, 10 (1) : 1
1 6, 5 (1) : 1
2 6, 6 (1) : 1
3 6, 7 (1) : 1
4 6, 8 (1) : 1
5 6, 9 (1) : 1
6 6, 10 (1) : 1
7 6, 11 (1) : 1
```

```
1 7, 6 (1) : 1
2 7, 7 (1) : 1
3 7, 8 (1) : 1
4 7, 9 (1) : 1
5 7, 10 (1) : 1
6 7, 11 (1) : 1
7 7, 11 (1) : 1
```

Step 2:

The **.fis** file is now to be loaded in the FIS editor to view the membership functions, the rules and the rule surface plot.

- On the command window of MATLAB **fuzzy** is typed to open the FIS editor.

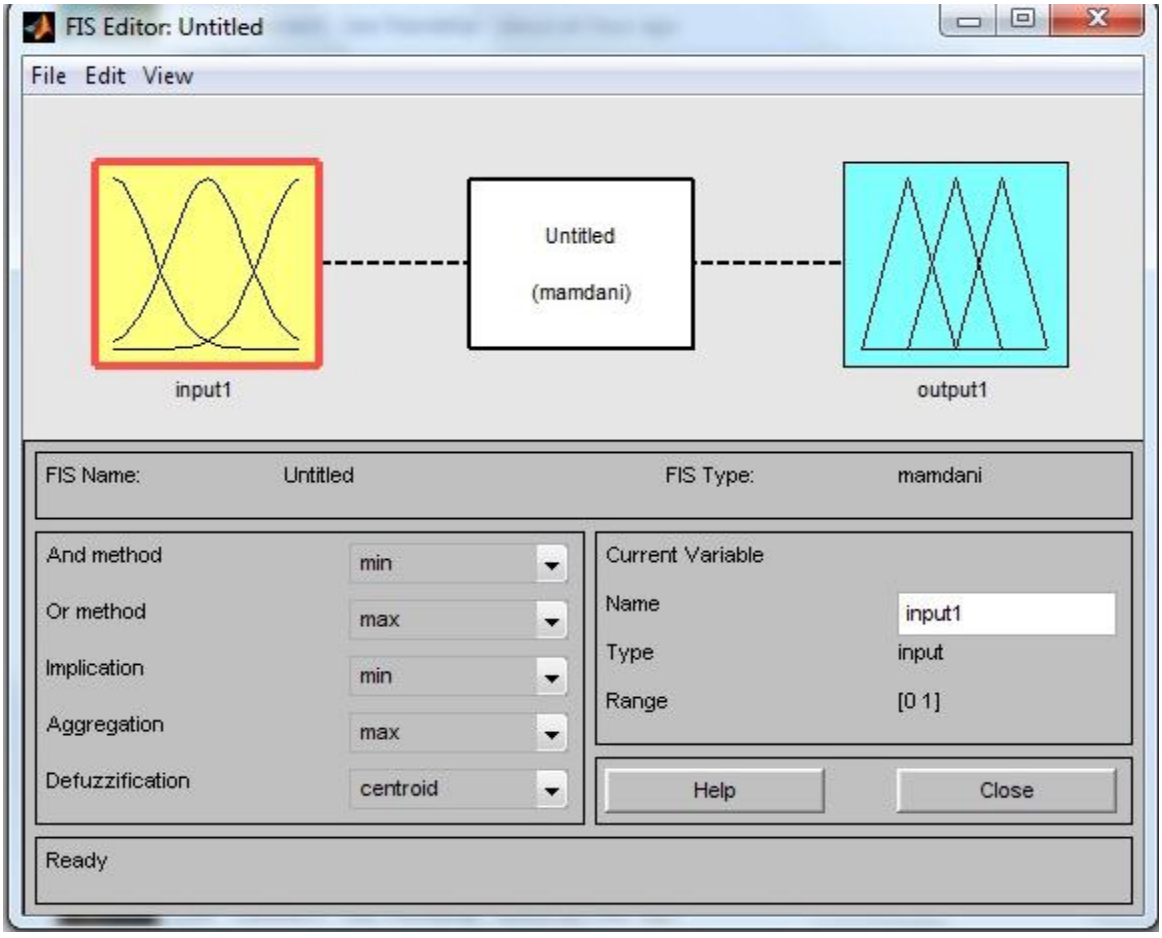


Fig 13: FIS editor window in MATLAB

- Click **File > Import > From file…** and then browse the **.fis** file to open **FIS editor: rules**.

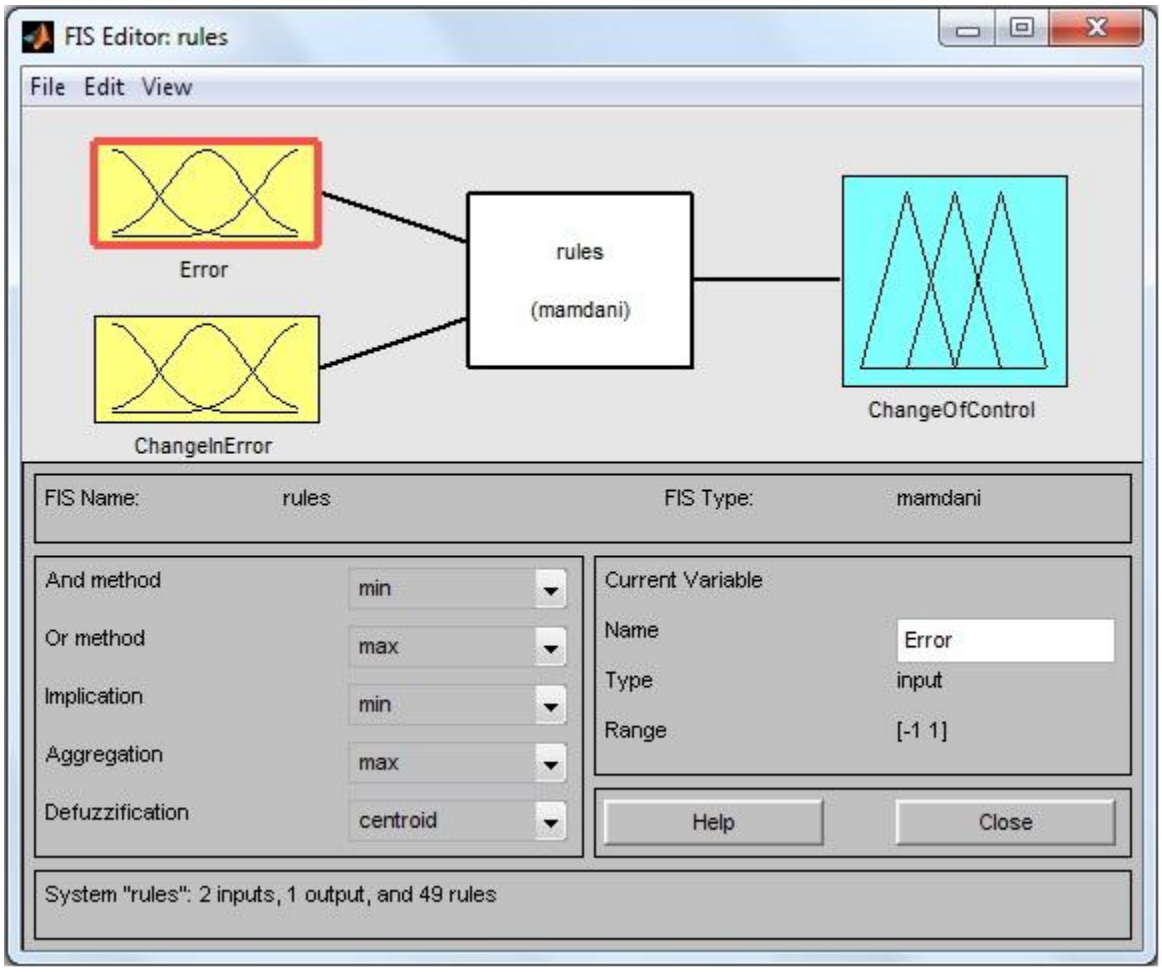


Fig 14: FIS editor: rules window in MATLAB

- Click on any of the input or output to view the respective membership functions. The membership functions for inputs Error and Change in Error and for output Change in Control are shown in Fig 15, Fig 16 and Fig 17 respectively.

- In the **FIS editor: rules** window click on **View > Surface** to view the three dimensional plot of the control surface. This plot is shown in Fig 18.

- Then in the **FIS editor: rules** window click on **View > Rules** to see the rules. The inputs can be changed in the window and respective outputs can be viewed.

Fig 19, Fig 20, Fig 21 and Fig 22 show outputs for four different inputs.

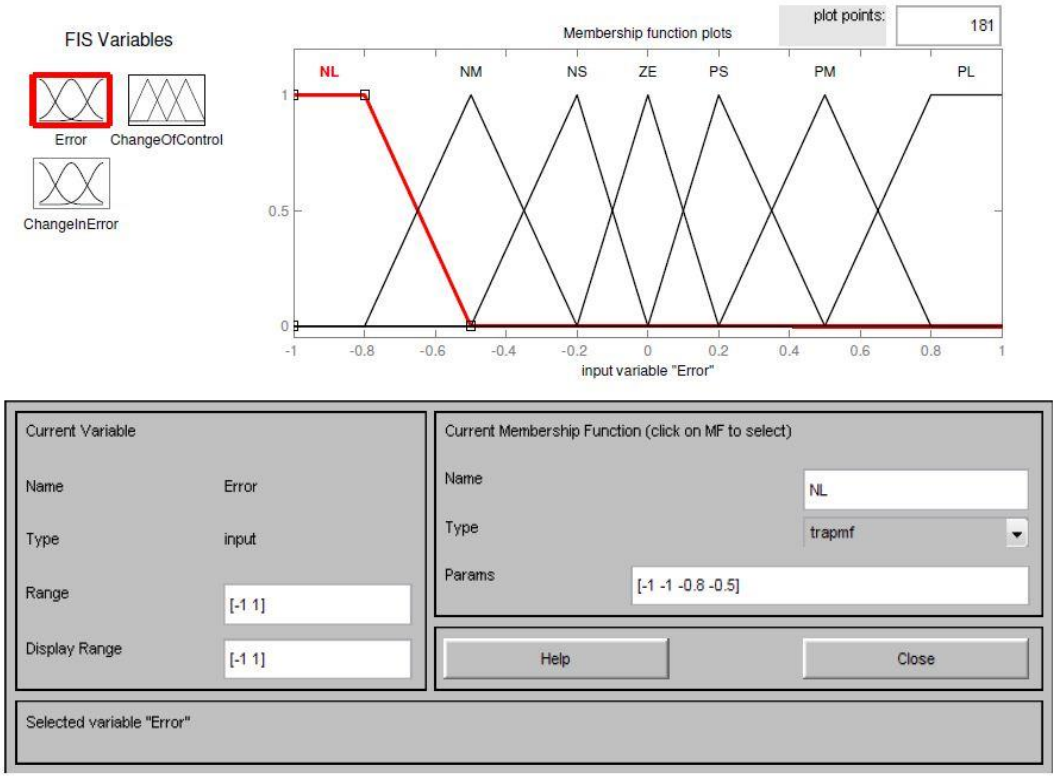


Fig 15: Membership function for the input Error (e)

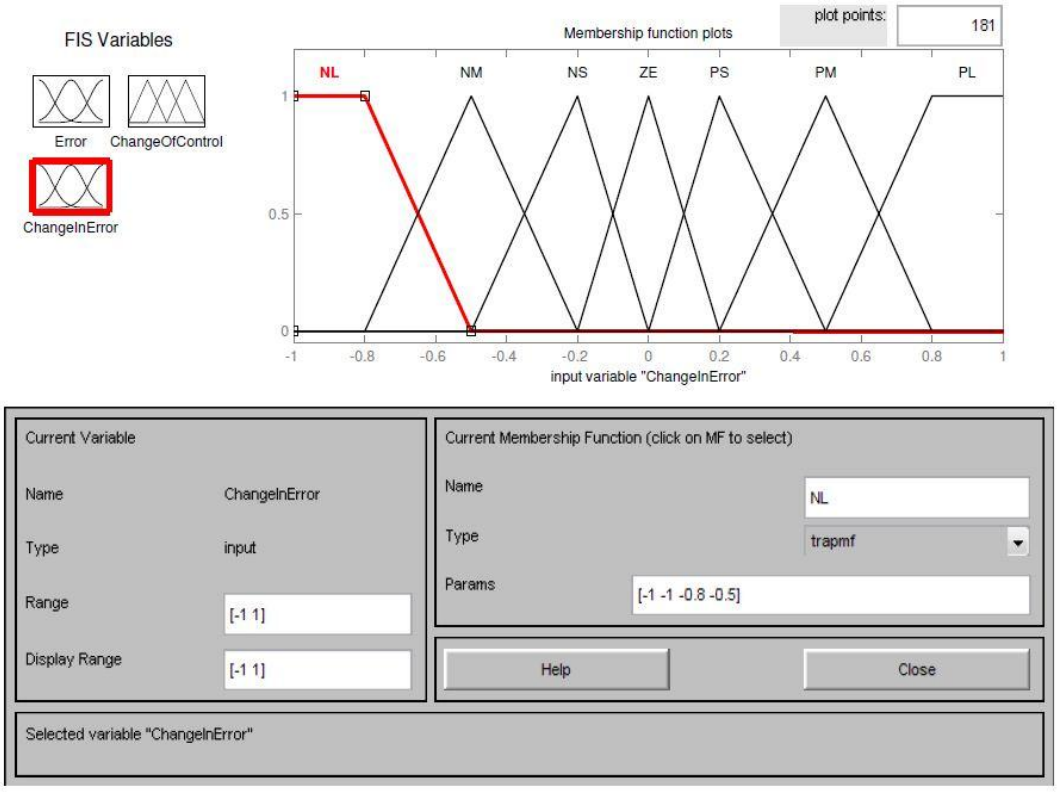


Fig 16: Membership function for the input Change in Error (Δe)

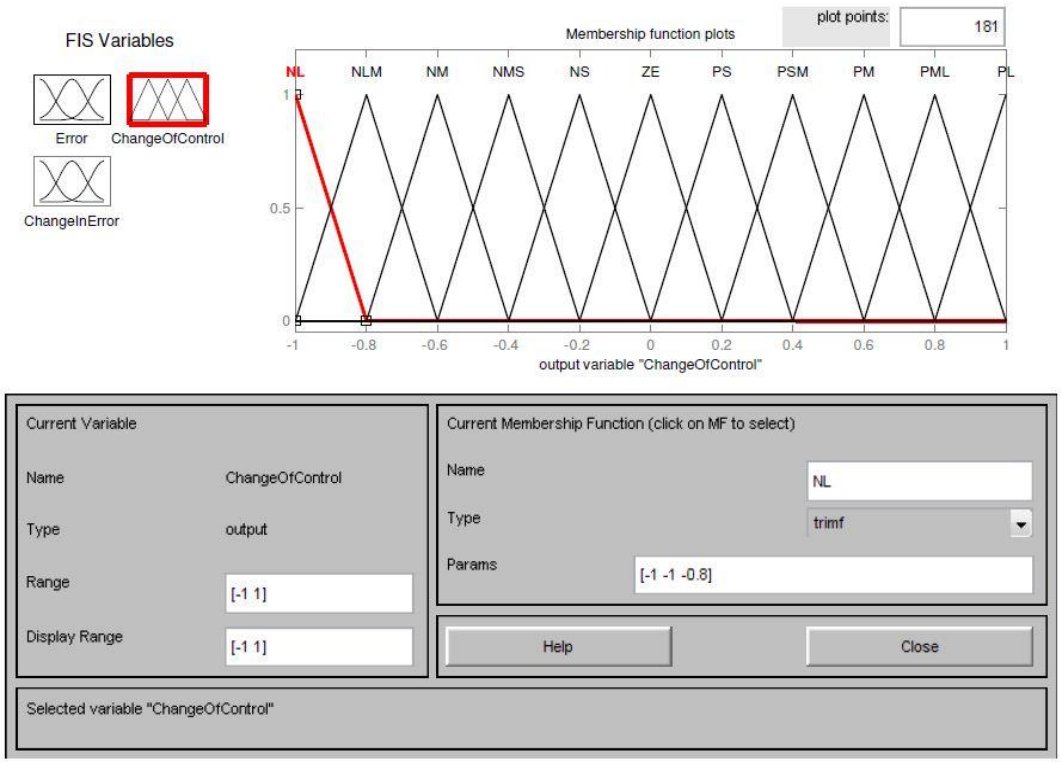


Fig 17: Membership function for the output Change of control (ω_{Sl})

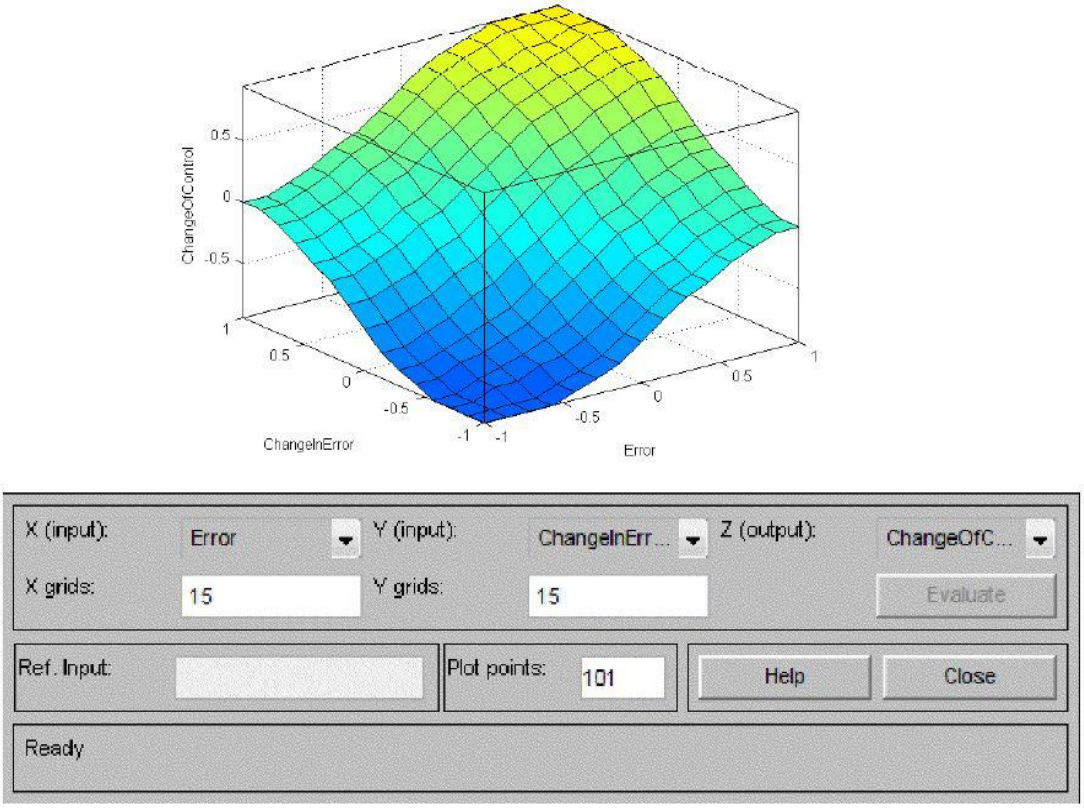


Fig 18: Three dimensional plot of the control surface

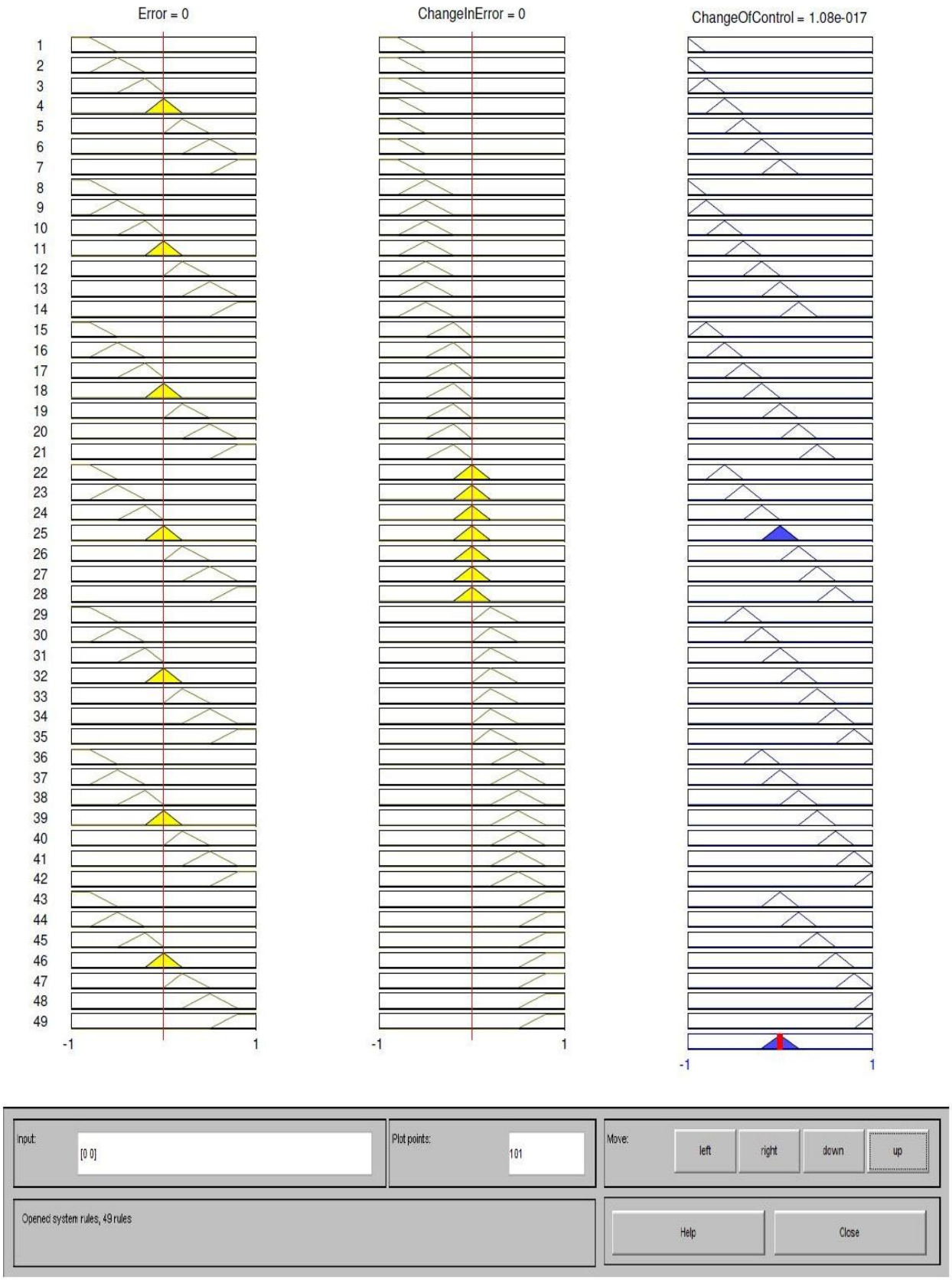


Fig 19: Rule viewer with inputs **e = 0** and **Δe = 0**

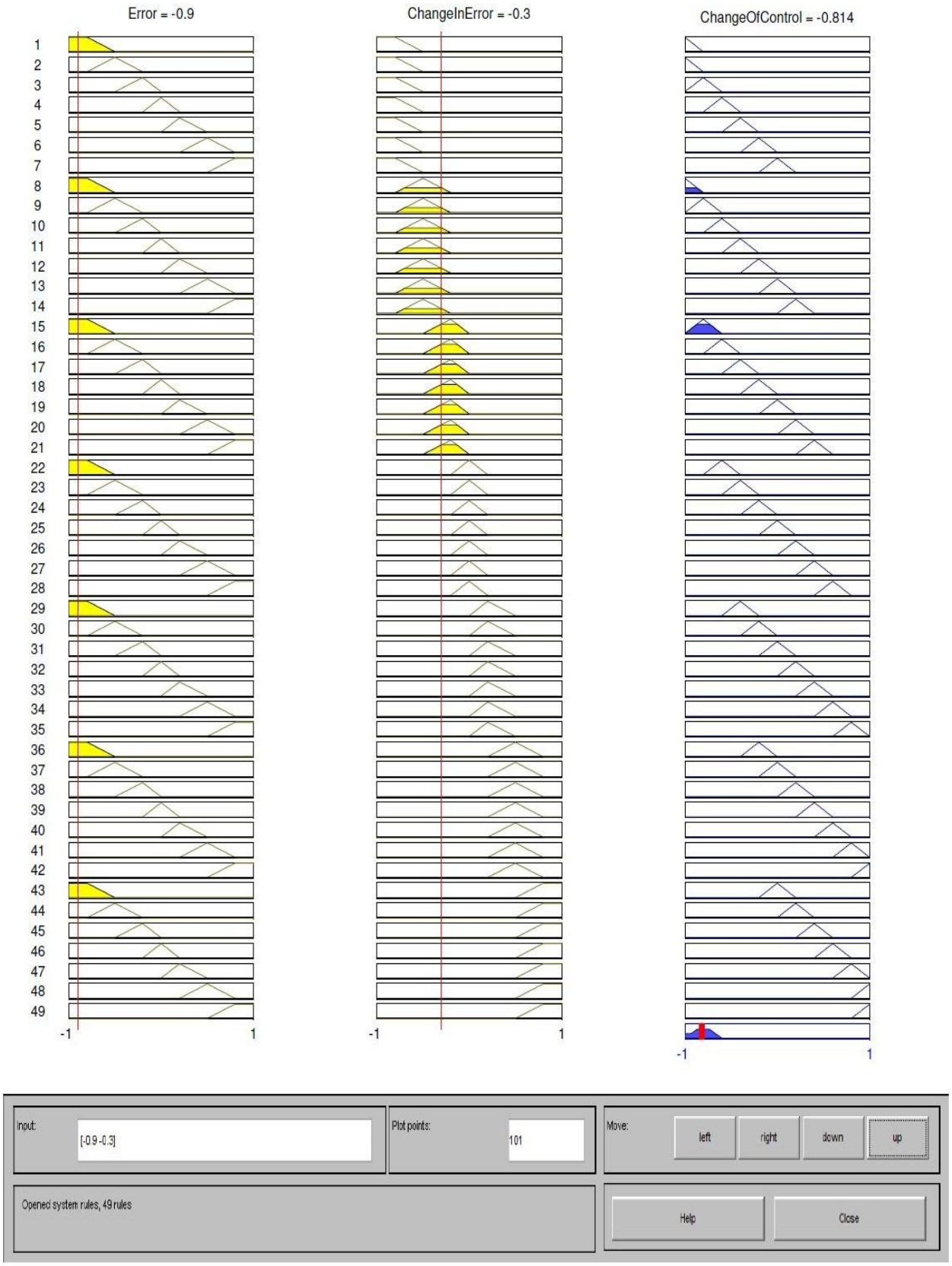


Fig 20: Rule viewer with inputs **e = -0.9** and **Δe = -0.3**

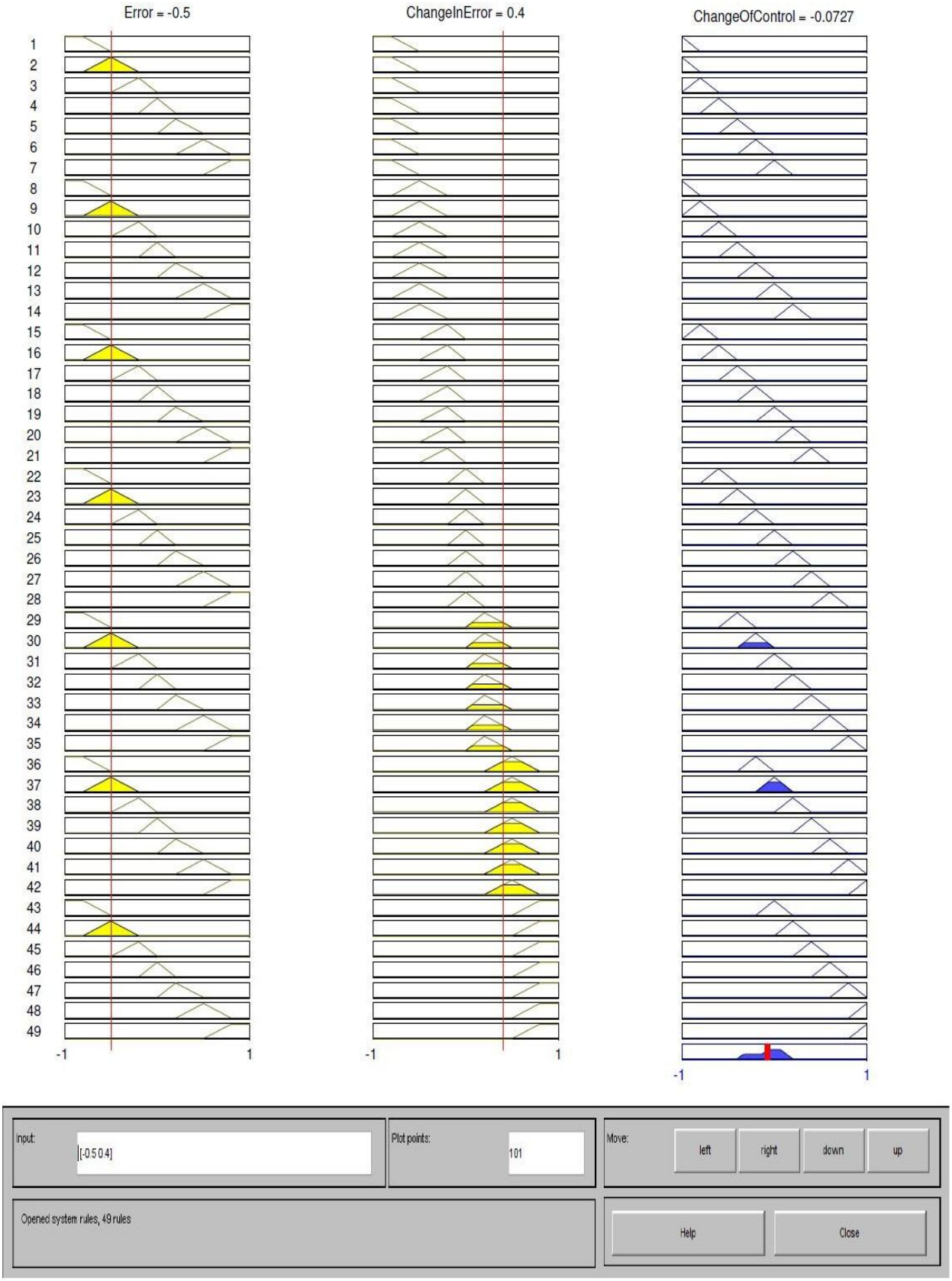


Fig 21: Rule viewer with inputs **e = -0.5** and **Δe = 0.4**

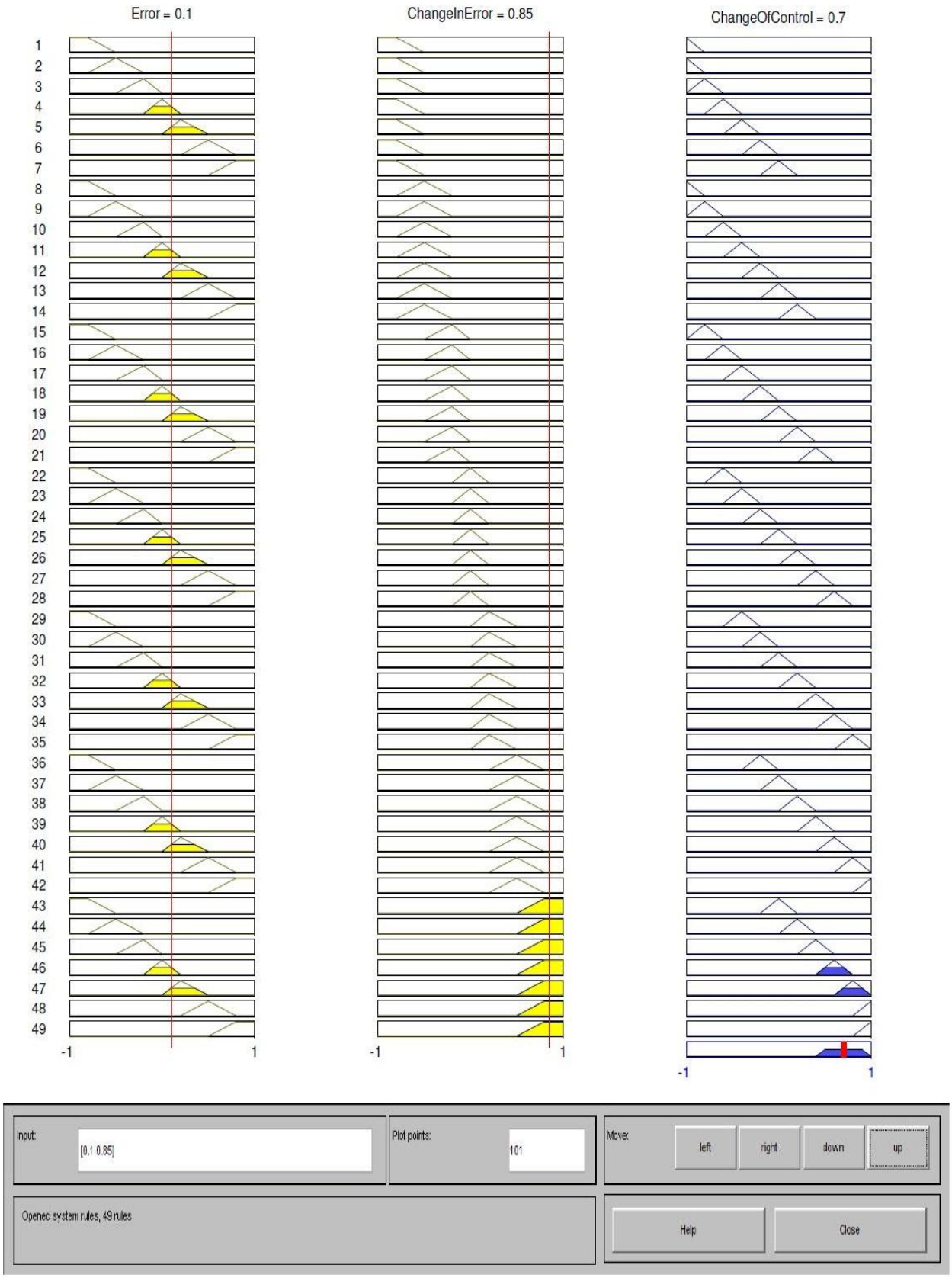


Fig 22: Rule viewer with inputs **e = 0.1** and **Δe = 0.85**

5.6 Summary

The Fuzzy Logic Controller is therefore designed and using the rule viewer the outputs are verified. Hence designing of the Fuzzy Logic Controller is now complete. This FIS file can now be loaded into the Fuzzy Logic Controller in the MATLAB/SIMULINK® and the simulation can be run. The controller can be refined accordingly to get the desired result during the simulation. Hence, we find that designing a Fuzzy Logic Controller using the Mamdani Fuzzy Model is quite convenient and does not require any cumbersome procedures.

Chapter 6

MATLAB Simulation

"The key to performance is elegance, not battalions of special cases"

(Jon Bentley, Doug Mcilroy)

6.1 Induction Motor Model in SIMULINK

The Induction Motor has been modeled in chapter 2, section 2.5. The very equations mentioned there can be used to build the Induction Motor model in SIMULINK. Fig. 23 shows block diagram of the induction motor with three inputs namely, input voltage (V_s), speed of induction motor (ω_m) and speed of the *dq* frame (ω_k). Torque and current are taken as outputs.

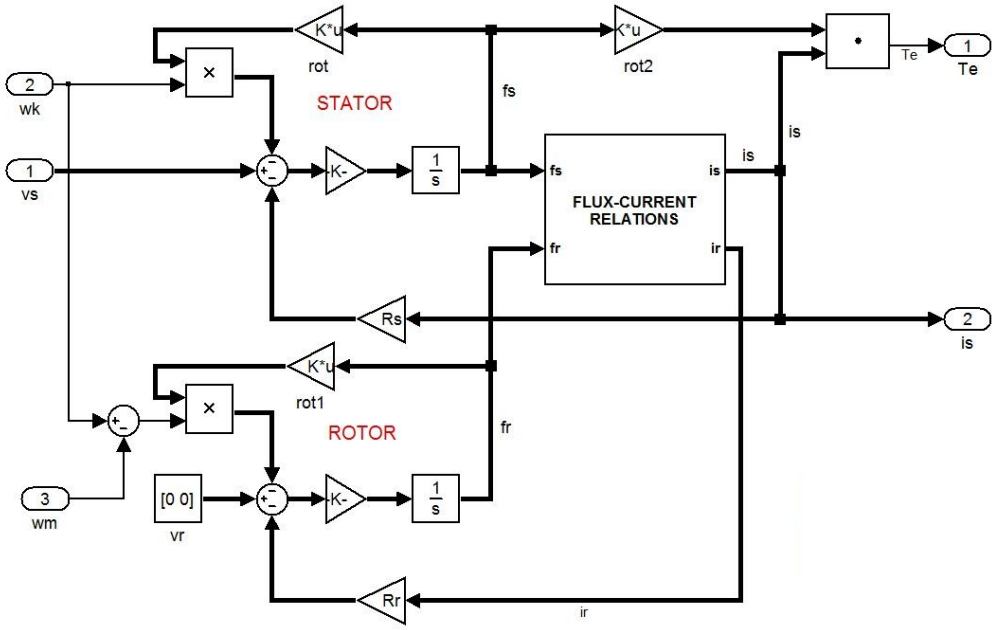


Fig 23: Space Vector Model of Induction Motor

The SIMULINK model for the Flux-Current Relations in the Fig 23 is shown in Fig 24. The mathematical model for the same is given in Chapter 2, Section 2.5.

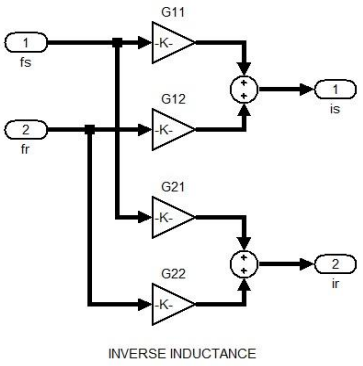


Fig 24: SIMULINK Model for Flux-Current Relations

The block diagram of scalar control of induction motor using fuzzy logic controller and PI controller designed in MATLAB/SIMULINK® is shown in Fig 25.

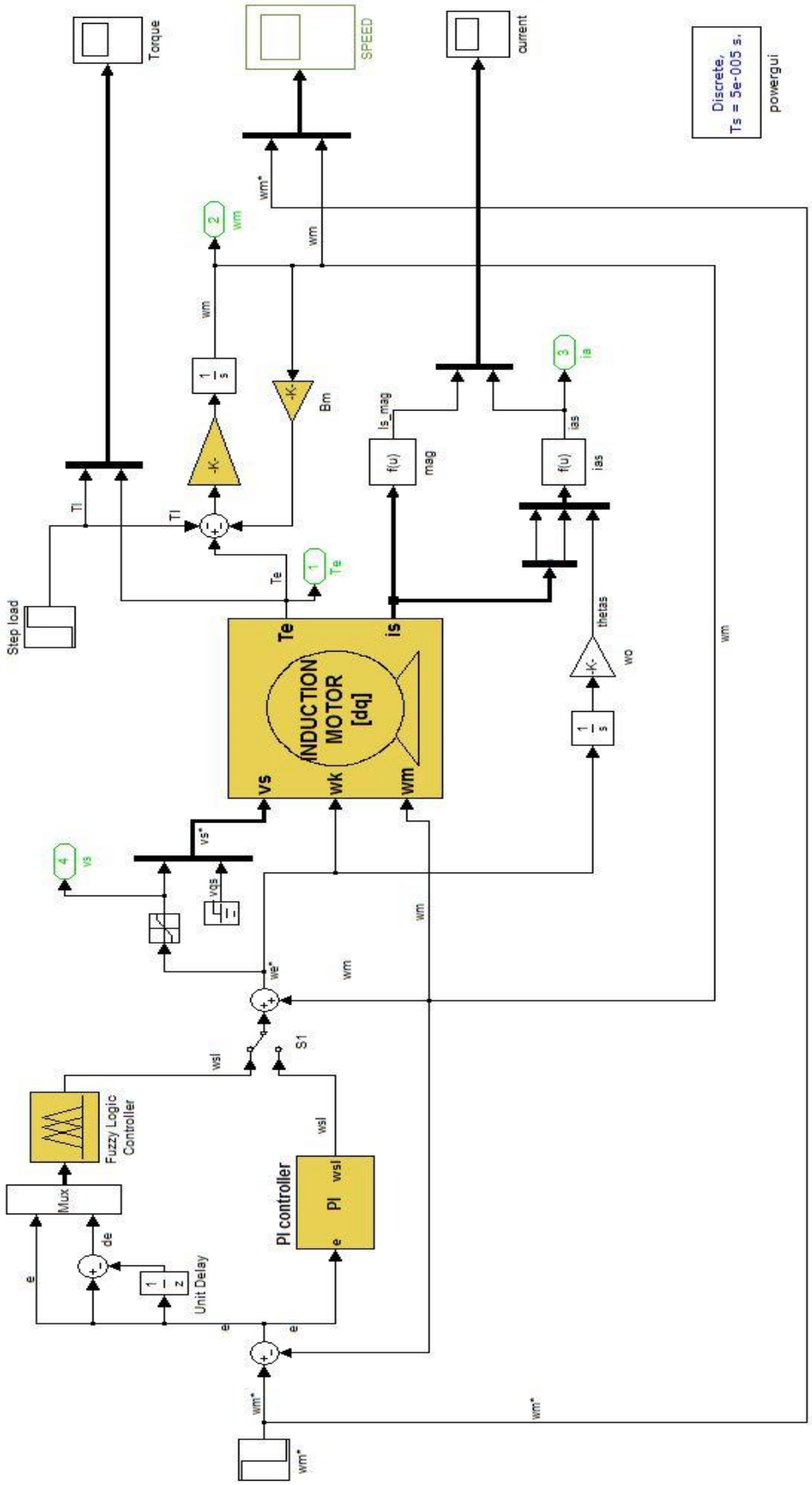


Fig 25: Block Diagram of Scalar Control of Induction Motor using Fuzzy Logic Controller and PI Controller

The parameters that have been used to describe the electrical and electromechanical systems are given below. These parameters are expressed in per unit (pu).

Parameter	Quantity
Stator Resistance	0.025 pu
Rotor Resistance	0.015 pu
Stator Leakage Inductance	0.10 pu
Rotor Leakage Inductance	0.01 pu
Magnetizing Inductance	3.0 pu
Base Frequency	2*π*50 rad/s
Number of Poles	2
Moment of Inertia	0.6 pu
Viscous Friction Coefficient	1e-5 pu

The parameters for the PI controller used for comparison with the Fuzzy Logic Controller are:

Parameter	Quantity
Proportional Constant	0.6
Integral Constant	5.6
Saturation Limit	0.03

6.2 Simulation Results

The block diagram in Fig 25 was simulated and the plots for speed, current, and torque using both Fuzzy Logic Controller and PI Controller were observed. These have been compared in the following pages.

Fig 26 and Fig 27 show the Speed versus Time plot with reference speed varying from 1 to 0.2pu for Fuzzy Logic Controller and PI Controller respectively. Fig 29 and Fig 30 show the Speed versus Time plot with reference speed varying from 1 to 0.2 pu for Fuzzy Logic Controller and PI Controller respectively. Fig 32 and Fig 33 show the Speed versus Time plot with reference speed varying from 1 to 0.2 pu for Fuzzy Logic Controller and PI Controller respectively. Fig 35 and Fig 36 show the Speed versus Time plot with reference speed varying from 1 to 0.2 pu for Fuzzy Logic Controller and PI Controller respectively.

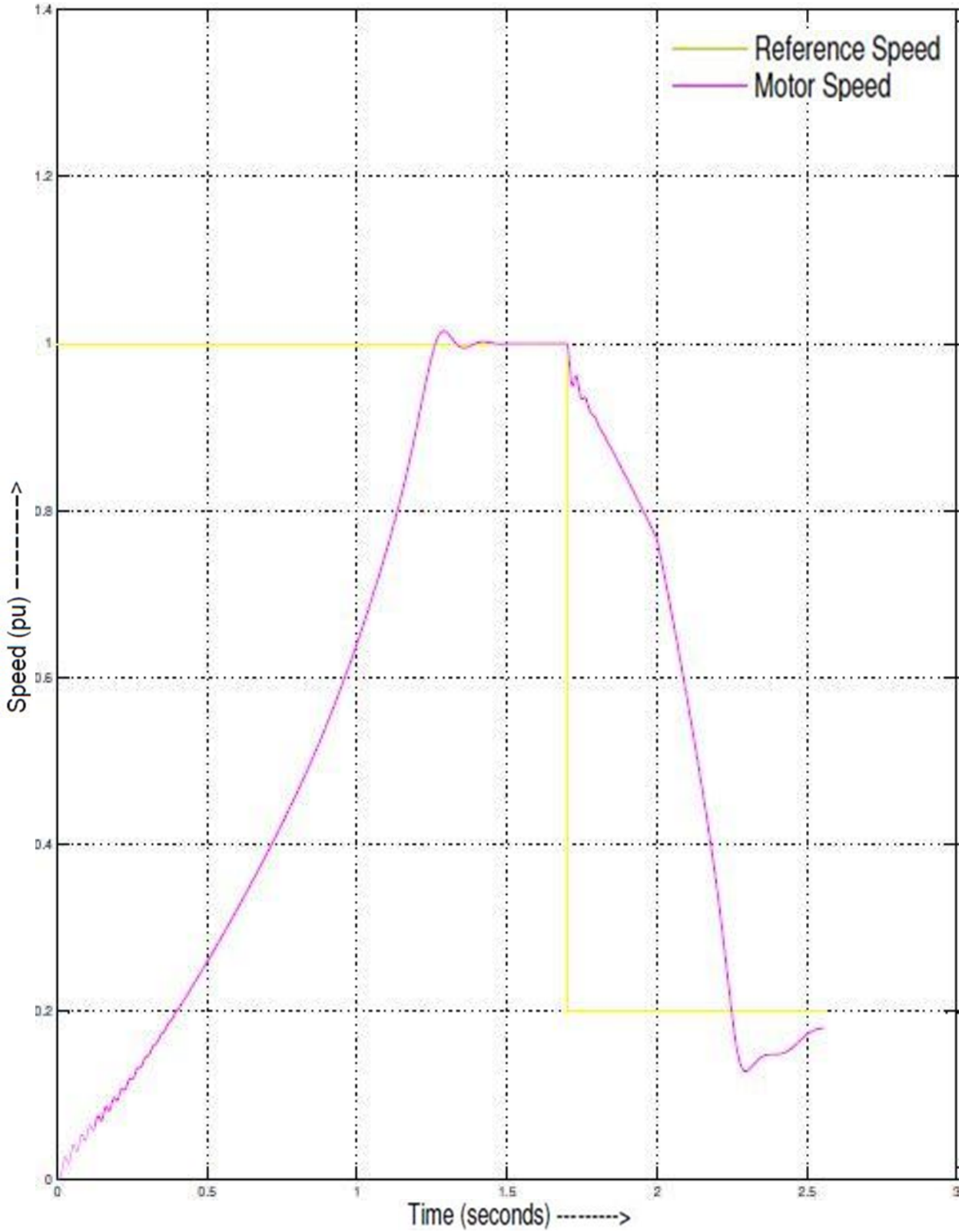


Fig 26: Speed versus Time Plot with Reference Speed (ω_m^*) varying from 1 to 0.2 pu using Fuzzy Logic Controller.

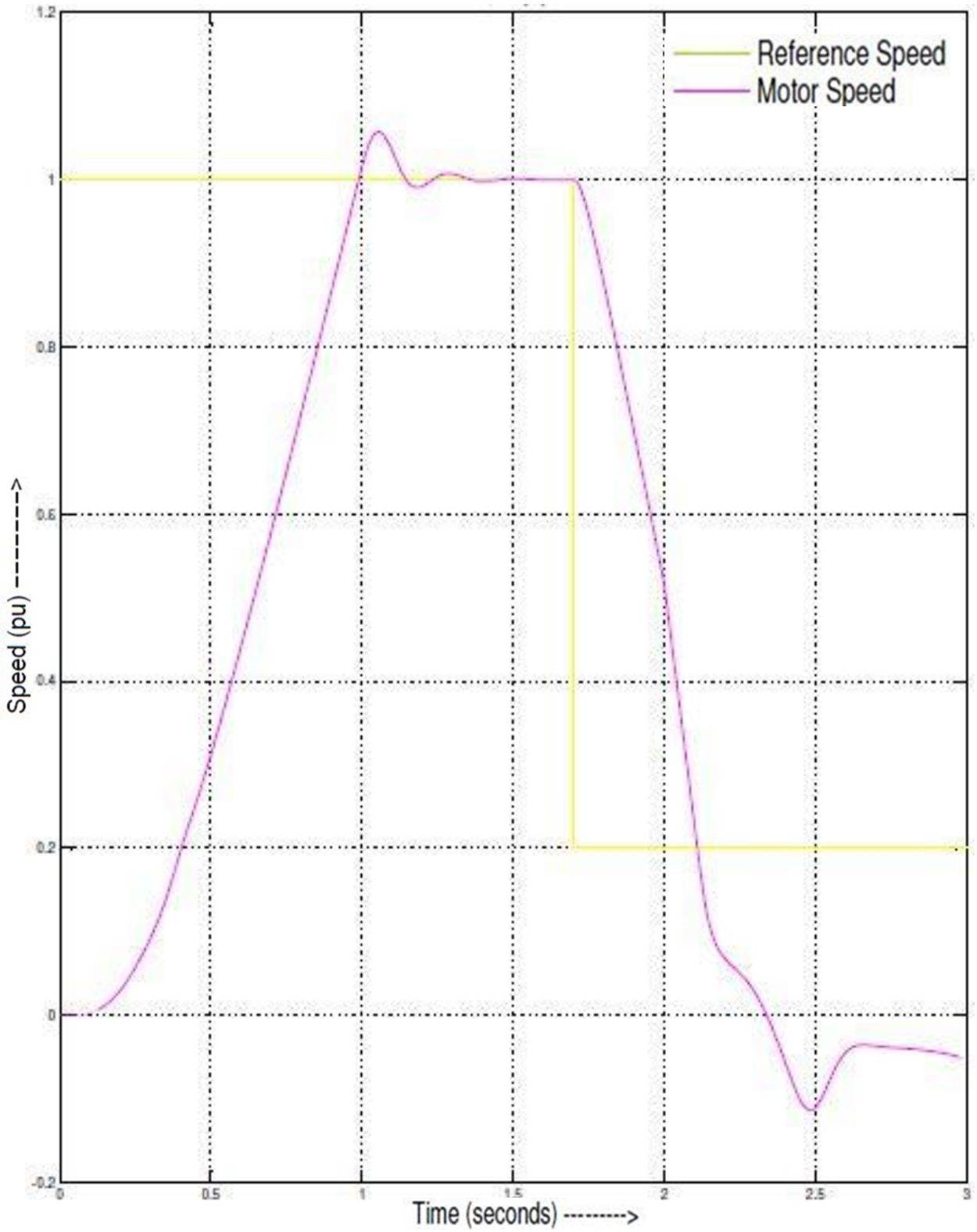


Fig 27: Speed versus Time Plot with Reference Speed (ω_m^*) varying from 1 to 0.2 pu using PI Controller

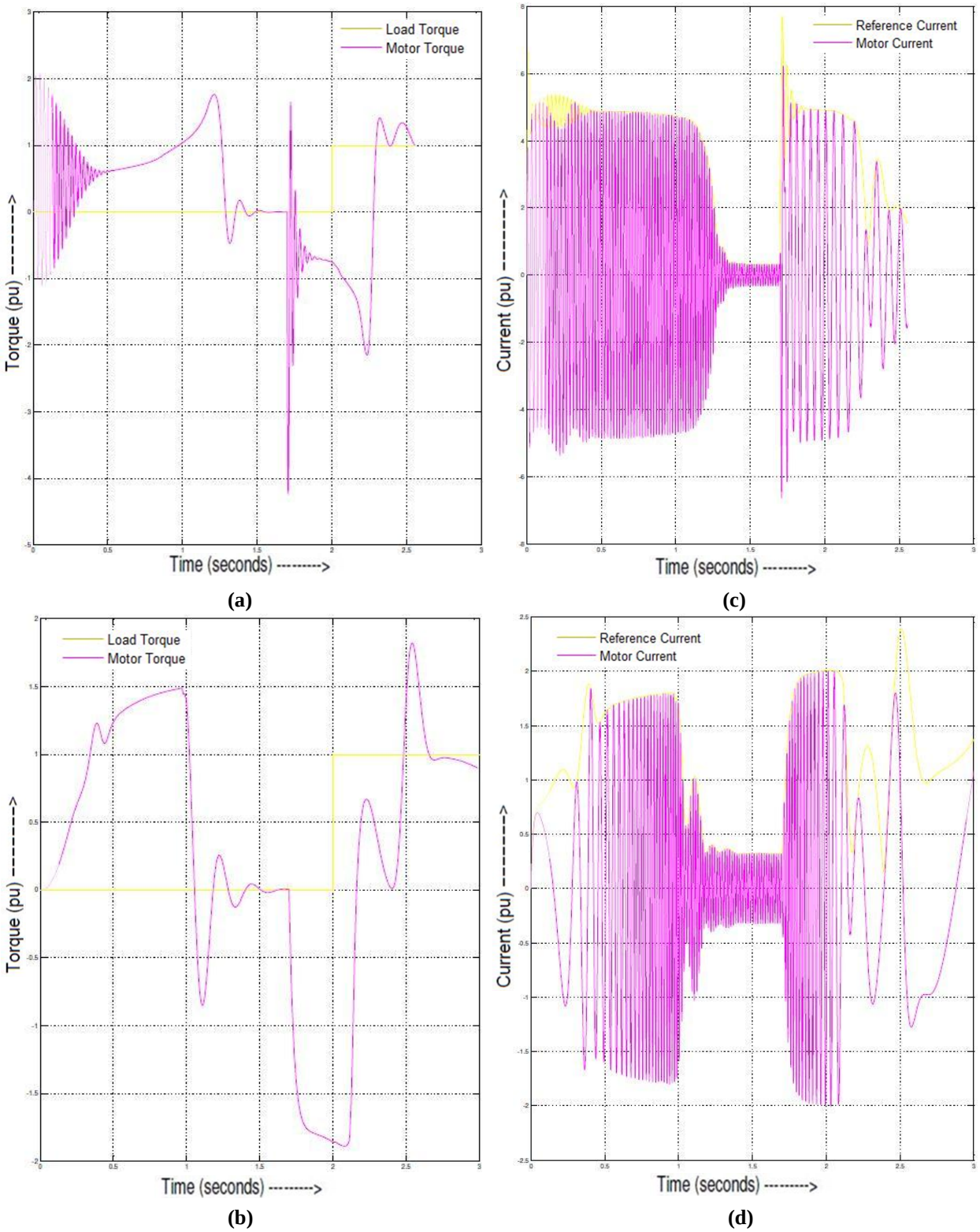


Fig 28: (a), **(b)** – Torque versus Time Plot with ω_m^* varying from 1 to 0.2 pu using FLC & PI Controller respectively.

(c), **(d)** – Current versus Time Plot with ω_m^* varying from 1 to 0.2 pu using FLC & PI Controller respectively.

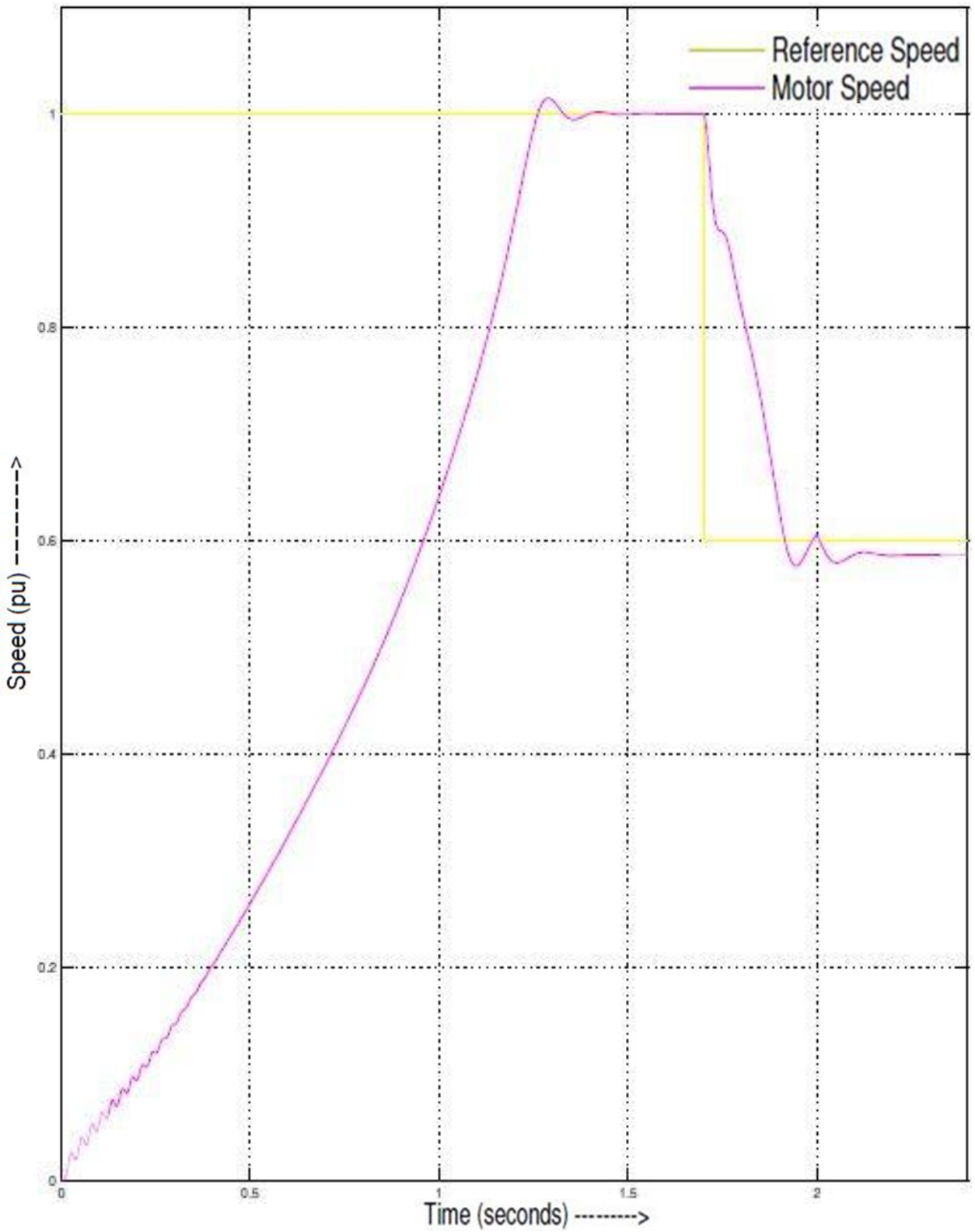


Fig 29: Speed versus Time Plot with Reference Speed (ω_m^*) varying from 1 to 0.6 pu using Fuzzy Logic Controller

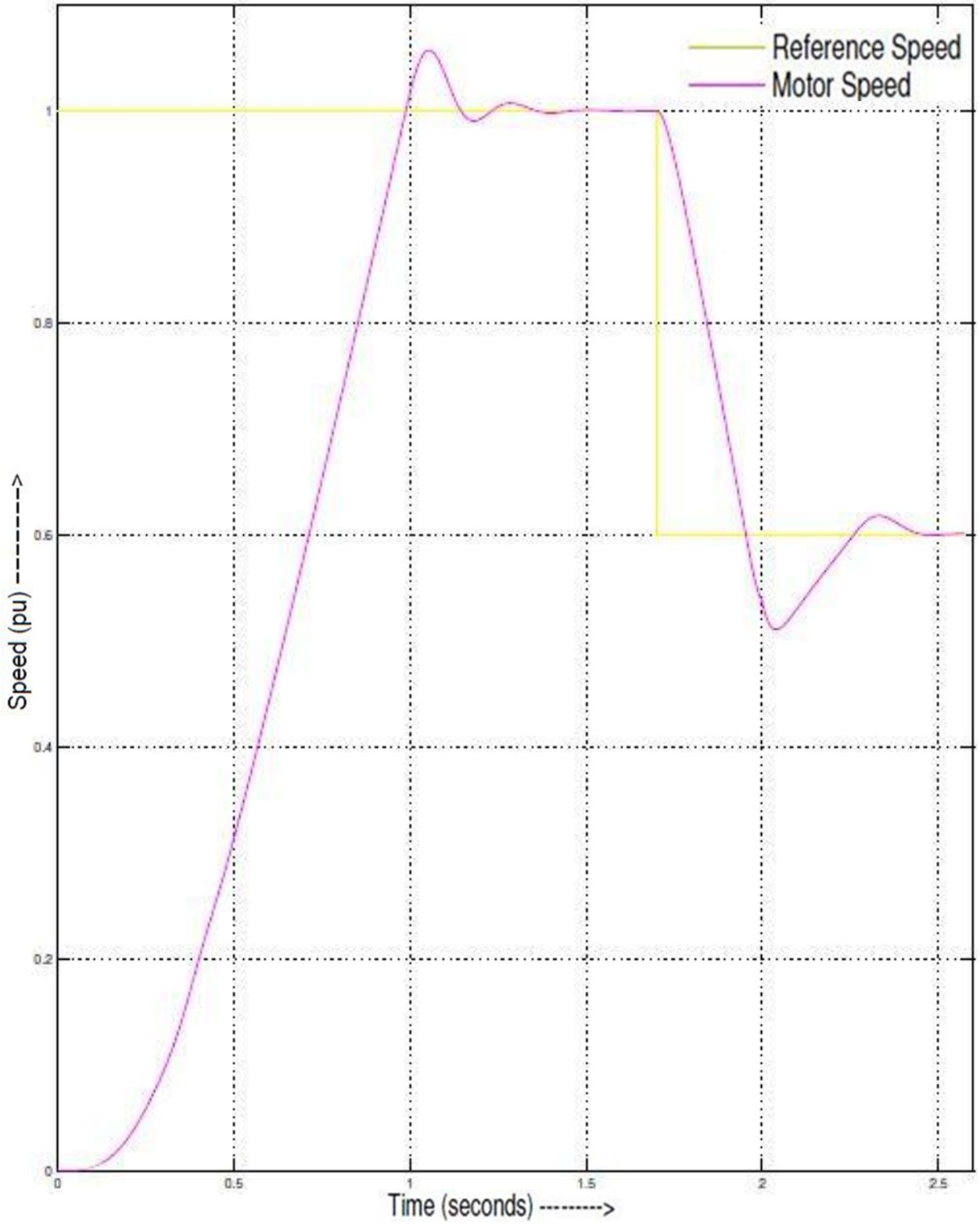


Fig 30: Speed versus Time Plot with Reference Speed (ω_m^*) varying from 1 to 0.6 pu using PI Controller.

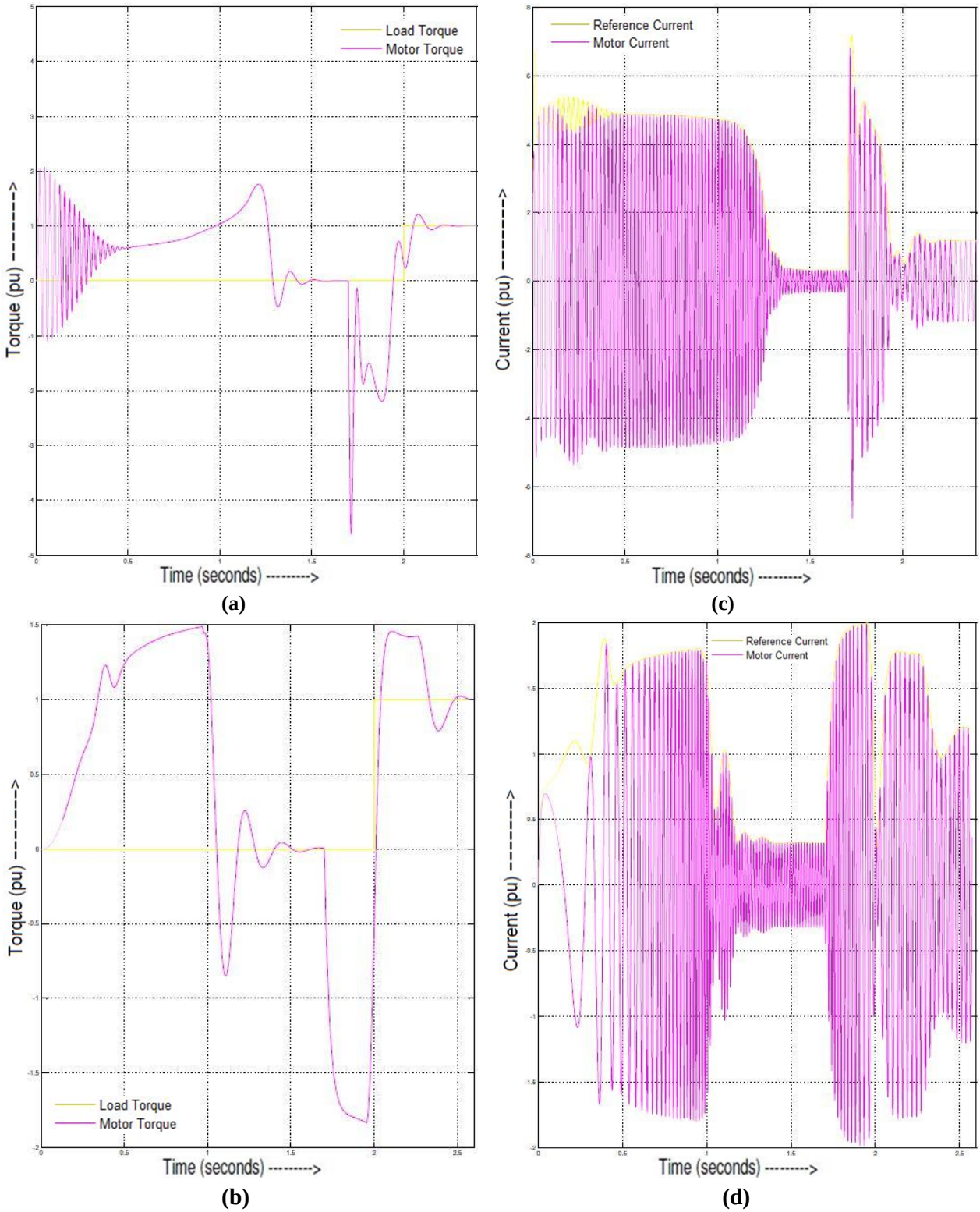


Fig 31: (a), (b) – Torque versus Time Plot with ω_m^* varying from 1 to 0.6 pu using FLC & PI Controller respectively.

(c), (d) – Current versus Time Plot with ω_m^* varying from 1 to 0.6 pu using FLC & PI Controller respectively.

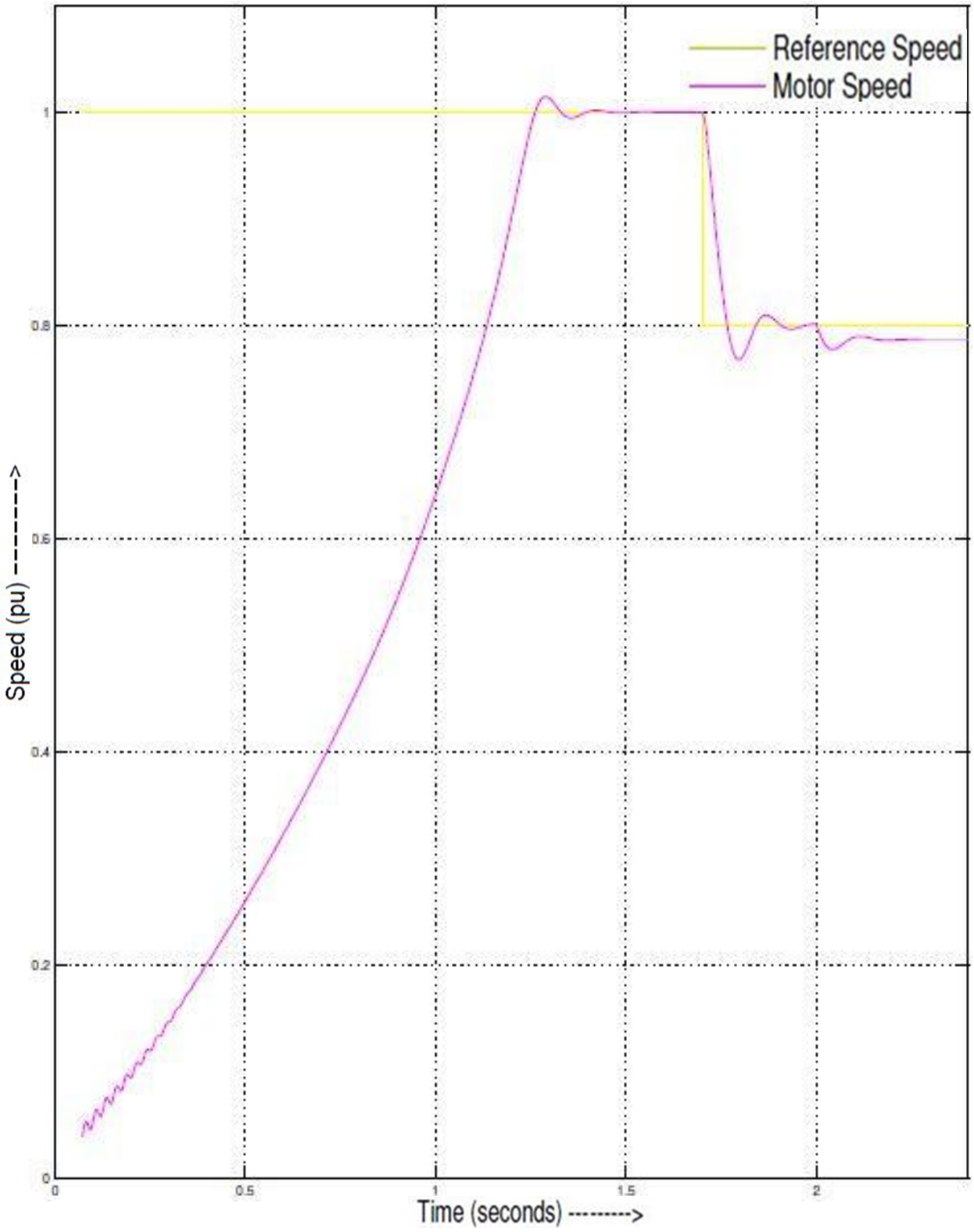


Fig 32: Speed versus Time Plot with Reference Speed (ω_m^*) varying from 1 to 0.8 pu using Fuzzy Logic Controller.

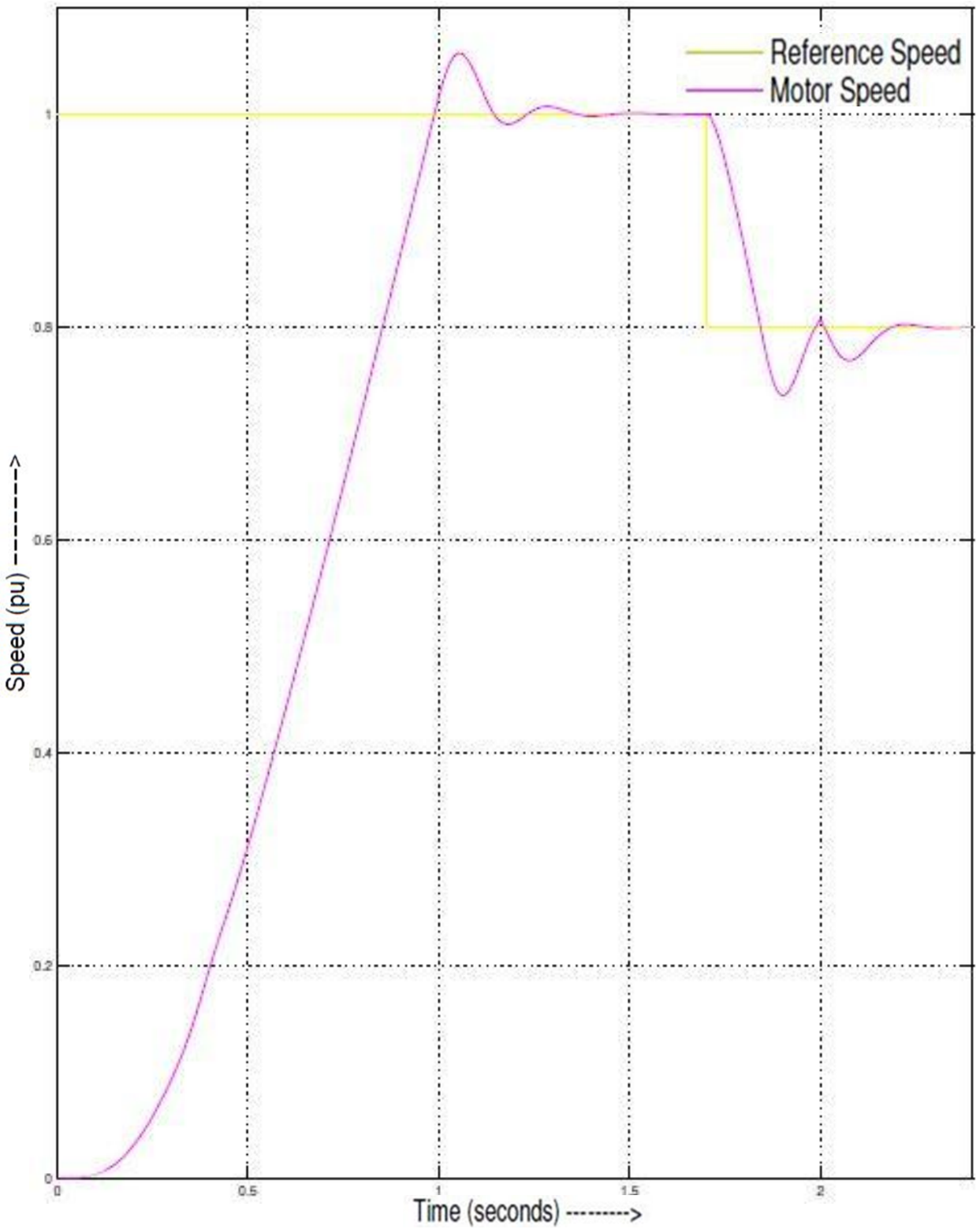


Fig 33: Speed versus Time Plot with Reference Speed (ω_m^*) varying from 1 to 0.8 pu using PI Controller.

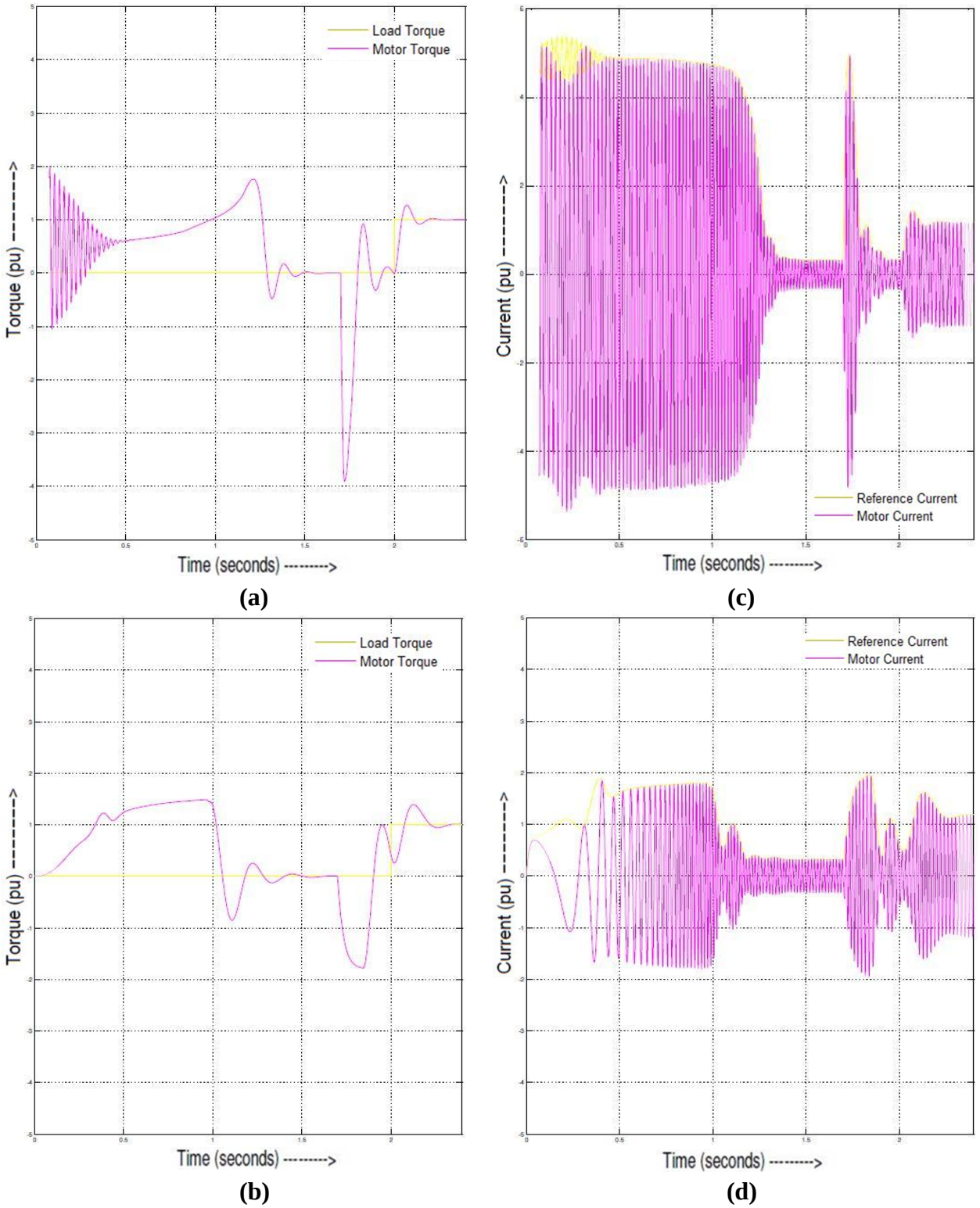


Fig 34: (a), (b) – Torque versus Time Plot with ω_m^* varying from 1 to 0.8 pu using FLC & PI Controller respectively.

(c), (d) – Current versus Time Plot with ω_m^* varying from 1 to 0.8 pu using FLC & PI Controller respectively.

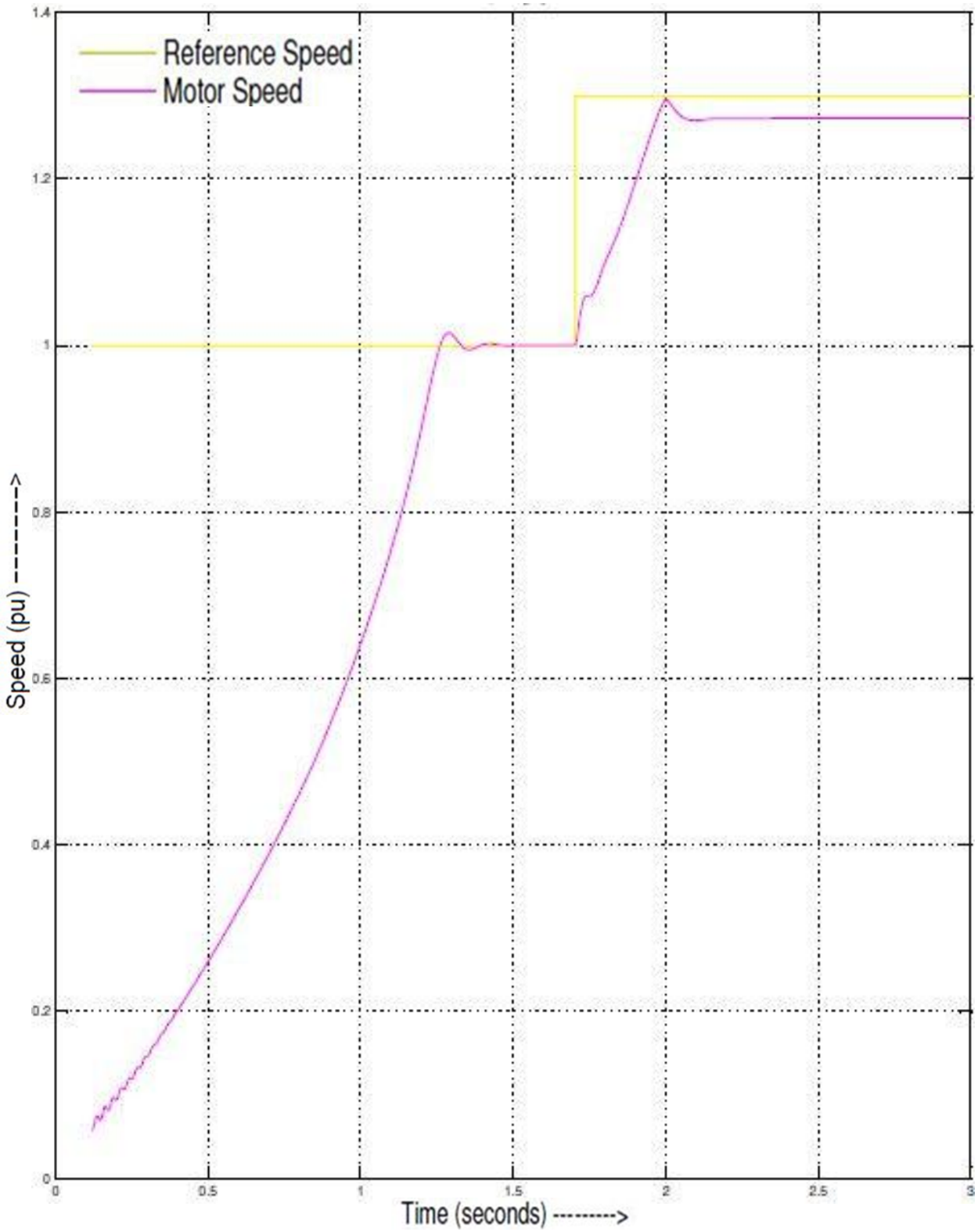


Fig 35: Speed versus Time Plot with Reference Speed (ω_m^*) varying from 1 to 1.3 pu using Fuzzy Logic Controller.

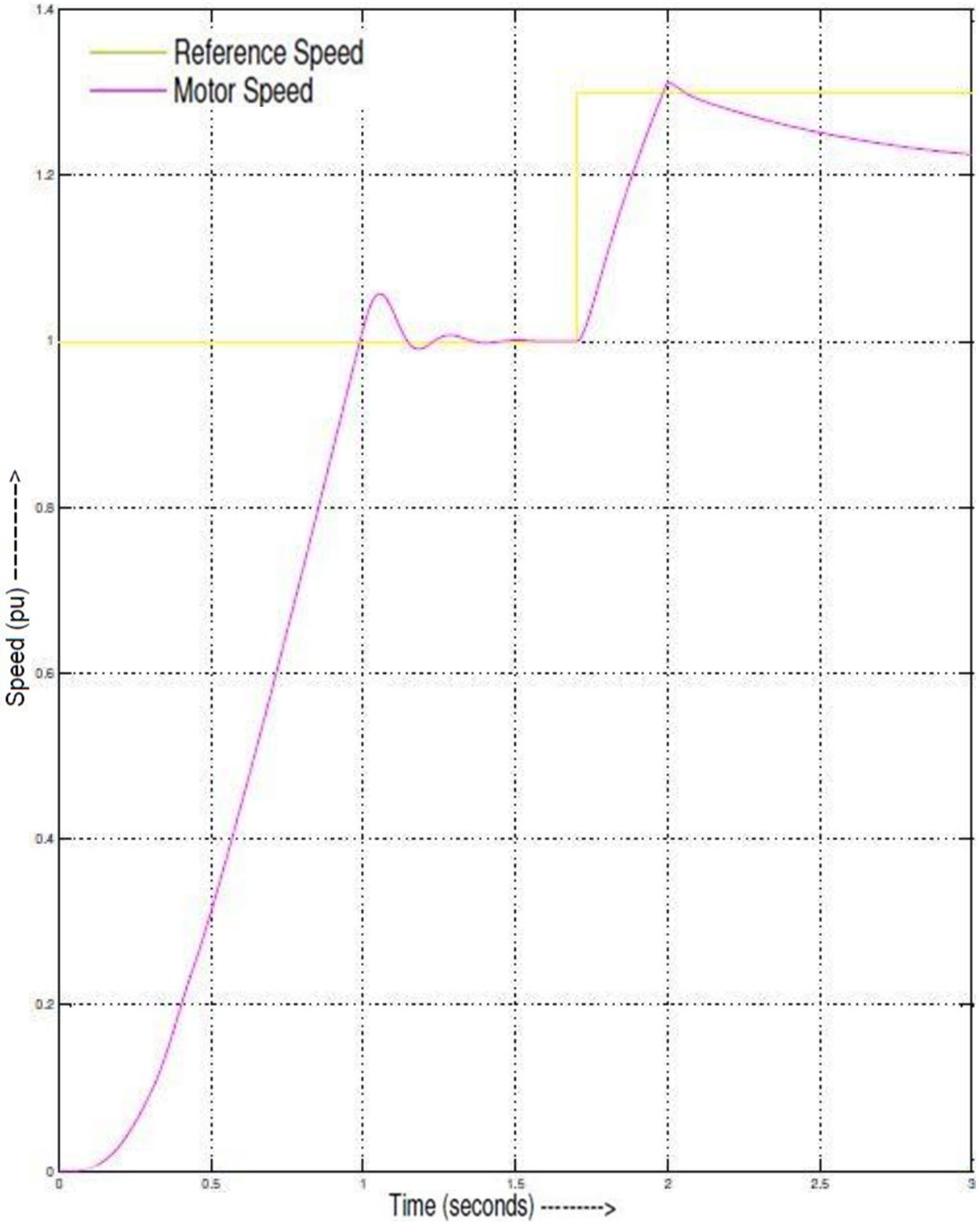
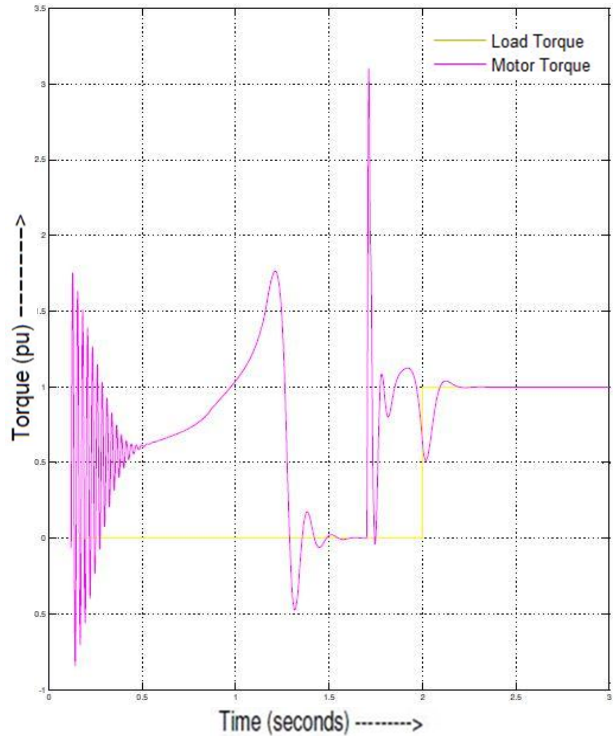
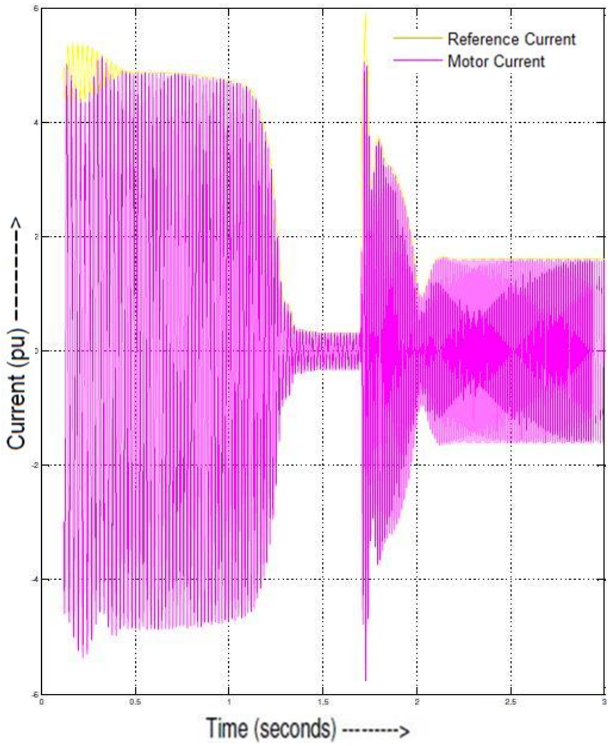


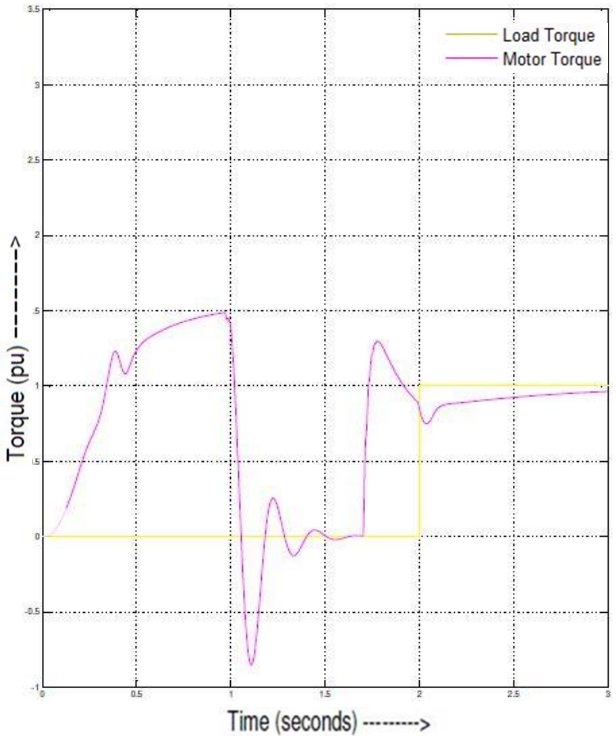
Fig 36: Speed versus Time Plot with Reference Speed (ω_m^*) varying from 1 to 1.3 pu using PI Controller.



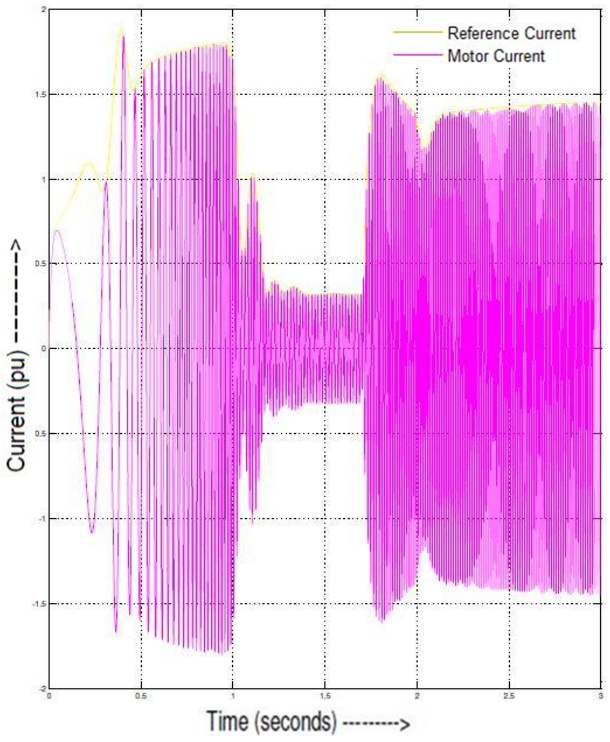
(a)



(c)



(b)



(d)

Fig 37: (a), (b) – Torque versus Time Plot with ω_m^* varying from 1 to 1.3 pu using FLC & PI Controller respectively.

(c), (d) – Current versus Time Plot with ω_m^* varying from 1 to 1.3 pu using FLC & PI Controller respectively.

6.3 Comparision between FLC and PI Controller Results

It can be seen from the above figures that while using the Fuzzy Logic Controller the overshoots obtained are lesser as compared to the case when the PI Controller is used. The settling time is also less in case of the Fuzzy Logic Controller, but the rise time is larger. The Fuzzy Logic Controller, however, portraits a better response when the reference speed is changed (either decreased or increased with respect to the base speed). It tends to approach the new reference speed faster and has, comparatively, a very low overshoot. It can be observed from Fig 27 and Fig 36 that the PI controller diverges from the new reference speed and does not attend a steady state when it is very less as compared to the base speed or greater than the base speed. The Fuzzy Logic Controller on the other hand attains a steady state. Even though this attained speed is not exactly equal to the new reference speed, it is very much close to it.

The torque plots show that while using the Fuzzy Logic Controller oscillations occur during starting while the PI controller doesn't show any such characteristic. This is because the Fuzzy Logic Controller is based on random knowledge of data. The machine provides a desirable response after some time as the controller first has to learn from or adjust according to the data provided by the user.

From the current plots, the same inferences can be achieved. We can see that in all the current plots the current is sinusoidal. But there is a distortion in the envelope before the machine attains steady state. The reason for this is that during starting the machine passes through the unstable region.

6.4 Conclusion

The Fuzzy Logic Controller used in this simulation has some drawbacks along with is advantages. But these disadvantages, viz. (i) achievement of only near to exact reference speed after change in reference speed and (ii) high rise time, can be reduced by refining the membership functions. In this simulation we have taken hybrid of trapezoidal and triangular membership functions for the inputs and triangular membership functions for the output. We can choose Gaussian membership functions for refining the control. Also the membership functions near the zero region can be made narrower and those towards the outside can be made comparatively wider. The tuning of the control will be taken up as the next step for the project.

CHAPTER 7

Tuning of the FLC and Simulations with Variations of Reference Speed and Load

"It is either coincidence piled on top of coincidence or it is deliberate design."

(Robert J. Sawyer)

7.1 Tuning of the Designed FLC

The Fuzzy Logic Controller, that has already been designed, was found to have certain drawbacks, namely (i) achievement of only near to exact reference speed after change in reference speed and (ii) high rise time. Out of these, the major drawback is the fact that the motor speed does not follow the reference speed unerringly. The presence of a steady state error makes it impossible for the Fuzzy Logic Controller to be used in any major application.

It can be inferred from the previous simulations that for a certain point or points in the input functions there are no rules defined. Hence, before going further with any other application the Fuzzy Logic Controller has to be tuned so that the steady state error is eliminated. The Controller was tuned by trial and error method. The same hybrid type membership functions have been used as earlier but the area of each membership function has been changed to get the desired results. It is known that to achieve finer control, the membership functions near the zero region should be made narrow and wider membership functions away from the zero region provides faster response to the system. Following this principle the membership functions have been modified. The tables and the figures provided in this chapter represent the tuned Fuzzy Logic Controller which completely eliminates the steady state error.

7.2 Modified Controller

7.2.1 Modified MFs for Input Variable Speed Error (e)

Table 5: Modified Fuzzy sets and the respective membership functions for speed error (e)

Fuzzy set or label	Set Description	Range	Membership Function
NL (Negative Large)	Speed error is high in the negative direction.	-1.0 to -1.0 -1.0 to -0.6 -0.6 to -0.3	Trapezoidal
NM (Negative Medium)	Speed error is medium in the negative direction.	-0.6 to -0.3 -0.3 to -0.03	Triangular
NS (Negative Small)	Speed error is small in the negative direction.	-0.3 to -0.125 -0.125 to 0	Triangular
ZE (Zero)	Speed error is around zero.	-0.03 to 0 0 to 0.03	Triangular
PS (Positive Small)	Speed error is small in the positive direction.	0 to 0.125 0.125 to 0.3	Triangular
PM (Positive Medium)	Speed error is medium in the positive direction.	0.03 to 0.3 0.3 to 0.6	Triangular
PL (Positive Large)	Speed error is high in the positive direction.	0.3 to 0.6 0.6 to 1.0 1.0 to 1.0	Trapezoidal

7.2.2 Modified MFs for Input Variable Change in Error (Δe)

Table 6: Modified Fuzzy sets and the respective membership functions for Change in Error (Δe)

Fuzzy set or label	Set Description	Range	Membership Function
NL (Negative Large)	Speed error is high in the negative direction.	-1.0 to -1.0 -1.0 to -0.6 -0.6 to -0.3	Trapezoidal
NM (Negative Medium)	Speed error is medium in the negative direction.	-0.6 to -0.3 -0.3 to -0.03	Triangular
NS (Negative Small)	Speed error is small in the negative direction.	-0.3 to -0.125 -0.125 to 0	Triangular
ZE (Zero)	Speed error is around zero.	-0.03 to 0 0 to 0.03	Triangular
PS (Positive Small)	Speed error is small in the positive direction.	0 to 0.125 0.125 to 0.3	Triangular
PM (Positive Medium)	Speed error is medium in the positive direction.	0.03 to 0.3 0.3 to 0.6	Triangular
PL (Positive Large)	Speed error is high in the positive direction.	0.3 to 0.6 0.6 to 1.0 1.0 to 1.0	Trapezoidal

7.2.3 Modified MFs for Output Variable Change of Control (ω_{sl})

Table 7: Modified Fuzzy sets and the respective MFs for Change of Control (ω_{sl})

Fuzzy set or Label	Range	Membership Function
NL (Negative Large)	-1.0 to -1.0 -1.0 to -0.8	Triangular
NLM (Negative Large Medium)	-1.0 to -0.8 -0.8 to -0.6	Triangular
NM (Negative Medium)	-0.8 to -0.6 -0.6 to -0.4	Triangular
NMS (Negative Medium Small)	-0.6 to -0.4 -0.4 to -0.2	Triangular
NS (Negative Small)	-0.4 to -0.2 -0.2 to 0	Triangular
ZE (Zero)	-0.2 to 0 0 to 0.2	Triangular
PS (Positive Small)	0 to 0.2 0.2 to 0.4	Triangular
PMS (Positive Medium Small)	0.2 to 0.4 0.4 to 0.6	Triangular

PM (Positive Medium)	0.4 to 0.6 0.6 to 0.8	Triangular
PLM (Positive Large Medium)	0.6 to 0.8 0.8 to 1.0	Triangular
PL (Positive Large)	0.8 to 1.0 1.0 to 1.0	Triangular

7.2.4 Modified Rule Base Design for the Output (ω_{sl})

The Rule Base for deciding the output of the inference system consists of 49 If-Then rules in this case since there are 7 fuzzy sets in each of the inputs. The table representing the rule base is as follows:

Table 8: Modified Fuzzy Rule Table for Output (ω_{sl})

Δe \ e	**NL**	**NM**	**NS**	**ZE**	**PS**	**PM**	**PL**
NL	NL	NL	NLM	NM	NMS	NS	ZE
NM	NL	NLM	NM	NMS	NS	ZE	PS
NS	NLM	NM	NMS	NS	ZE	PS	PMS
ZE	NM	NMS	NS	ZE	PS	PMS	PM
PS	NMS	NS	ZE	PS	PMS	PM	PLM
PM	NS	ZE	PS	PMS	PM	PLM	PL
PL	ZE	PS	PMS	PM	PLM	PL	PL

It can be seen that the membership functions for the output i.e. Change of Control (ω_{sl}) and the rule base of the inference system for the modified Fuzzy Logic Controller have not been changed.

7.2.5 Designing of the Modified Controller

The controller is again designed using the FIS editor in MATLAB/SIMULINK®. The same steps, as given in Chapter 5, were followed in the designing of the tuned controller. Trial and error method was used to modify the controller. A number of different arrangements of the membership functions were simulated one after the other and it was found that this particular arrangement provided a zero steady state error. Hence, this controller can be used for further applications.

The code for the modified Fuzzy Logic Controller is as follows:

```
[System]
Name='rules'
Type='mamdani'
Version=2.0
NumInputs=2
NumOutputs=1
NumRules=49
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'

[Input1]
Name='Error'
Range=[-1 1]
NumMFs=7
MF1='NL':'trapmf',[-1 -1 -0.6 -0.3]
MF2='NM':'trimf',[-0.6 -0.3 -0.03]
MF3='NS':'trimf',[-0.3 -0.125 0]
MF4='ZE':'trimf',[-0.03 0 0.03]
MF5='PS':'trimf',[0 0.125 0.3]
MF6='PM':'trimf',[0.03 0.3 0.6]
MF7='PL':'trapmf',[0.3 0.6 1 1]

[Input2]
Name='ChangeInError'
Range=[-1 1]
NumMFs=7
MF1='NL':'trapmf',[-1 -1 -0.6 -0.3]
MF2='NM':'trimf',[-0.6 -0.3 -0.03]
MF3='NS':'trimf',[-0.3 -0.125 0]
MF4='ZE':'trimf',[-0.03 0 0.03]
MF5='PS':'trimf',[0 0.125 0.3]
MF6='PM':'trimf',[0.03 0.3 0.6]
```

```
MF7='PL':'trapmf',[0.3 0.6 1 1]

[Output1]
Name='ChangeOfControl'
Range=[-1 1]
NumMFs=11
MF1='NL':'trimf',[-1 -1 -0.8]
MF2='NLM':'trimf',[-1 -0.8 -0.6]
MF3='NM':'trimf',[-0.8 -0.6 -0.4]
MF4='NMS':'trimf',[-0.6 -0.4 -0.2]
MF5='NS':'trimf',[-0.4 -0.2 0]
MF6='ZE':'trimf',[-0.2 0 0.2]
MF7='PS':'trimf',[0 0.2 0.4]
MF8='PSM':'trimf',[0.2 0.4 0.6]
MF9='PM':'trimf',[0.4 0.6 0.8]
MF10='PML':'trimf',[0.6 0.8 1]
MF11='PL':'trimf',[0.8 1 1]

[Rules]
1 1, 1 (1) : 1
2 1, 1 (1) : 1
3 1, 2 (1) : 1
4 1, 3 (1) : 1
5 1, 4 (1) : 1
6 1, 5 (1) : 1
7 1, 6 (1) : 1
1 2, 1 (1) : 1
2 2, 2 (1) : 1
3 2, 3 (1) : 1
4 2, 4 (1) : 1
5 2, 5 (1) : 1
6 2, 6 (1) : 1
7 2, 7 (1) : 1
1 3, 2 (1) : 1
2 3, 3 (1) : 1
3 3, 4 (1) : 1
4 3, 5 (1) : 1
```

```
5 3, 6 (1) : 1
6 3, 7 (1) : 1
7 3, 8 (1) : 1
1 4, 3 (1) : 1
2 4, 4 (1) : 1
3 4, 5 (1) : 1
4 4, 6 (1) : 1
5 4, 7 (1) : 1
6 4, 8 (1) : 1
7 4, 9 (1) : 1
1 5, 4 (1) : 1
2 5, 5 (1) : 1
3 5, 6 (1) : 1
4 5, 7 (1) : 1
5 5, 8 (1) : 1
6 5, 9 (1) : 1
7 5, 10 (1) : 1
1 6, 5 (1) : 1
2 6, 6 (1) : 1
3 6, 7 (1) : 1
4 6, 8 (1) : 1
5 6, 9 (1) : 1
6 6, 10 (1) : 1
7 6, 11 (1) : 1
1 7, 6 (1) : 1
2 7, 7 (1) : 1
3 7, 8 (1) : 1
4 7, 9 (1) : 1
5 7, 10 (1) : 1
6 7, 11 (1) : 1
7 7, 11 (1) : 1
```

The **.fis** file with the above program was loaded in the FIS Editor in MATLAB/SIMULINK®. Following the same steps as shown in Chapter 5 the membership functions, control surface and the rules were viewed. The figures given below show the modified membership functions for the inputs and the output, the modified control surface and the rule viewer with the same inputs, as chosen before, for the sake of comparison.

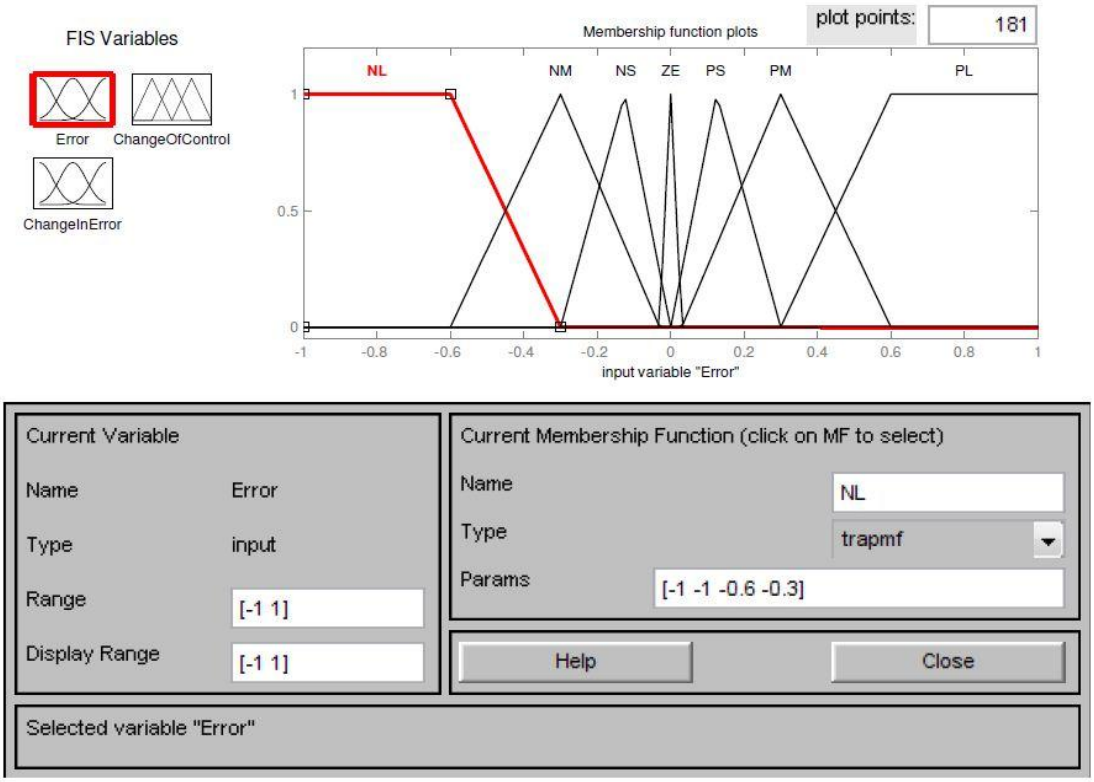


Fig 38: Modified membership functions for the input Error (e)

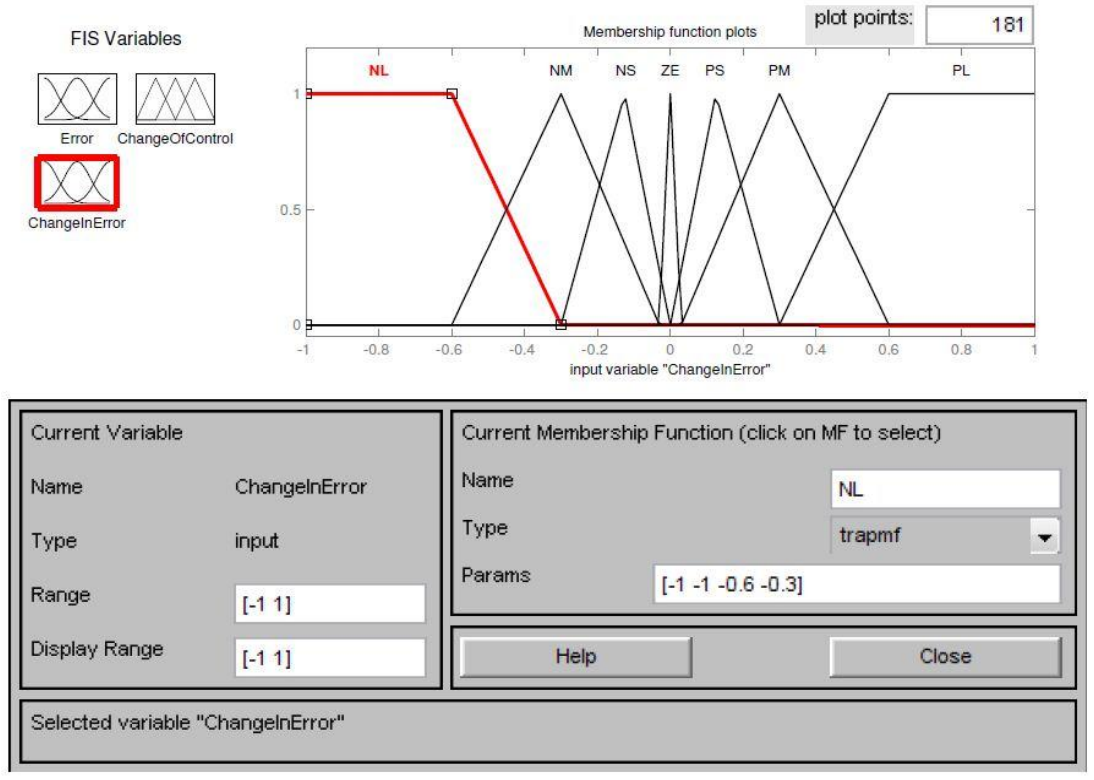


Fig 39: Modified membership functions for the input Change in Error (Δe)

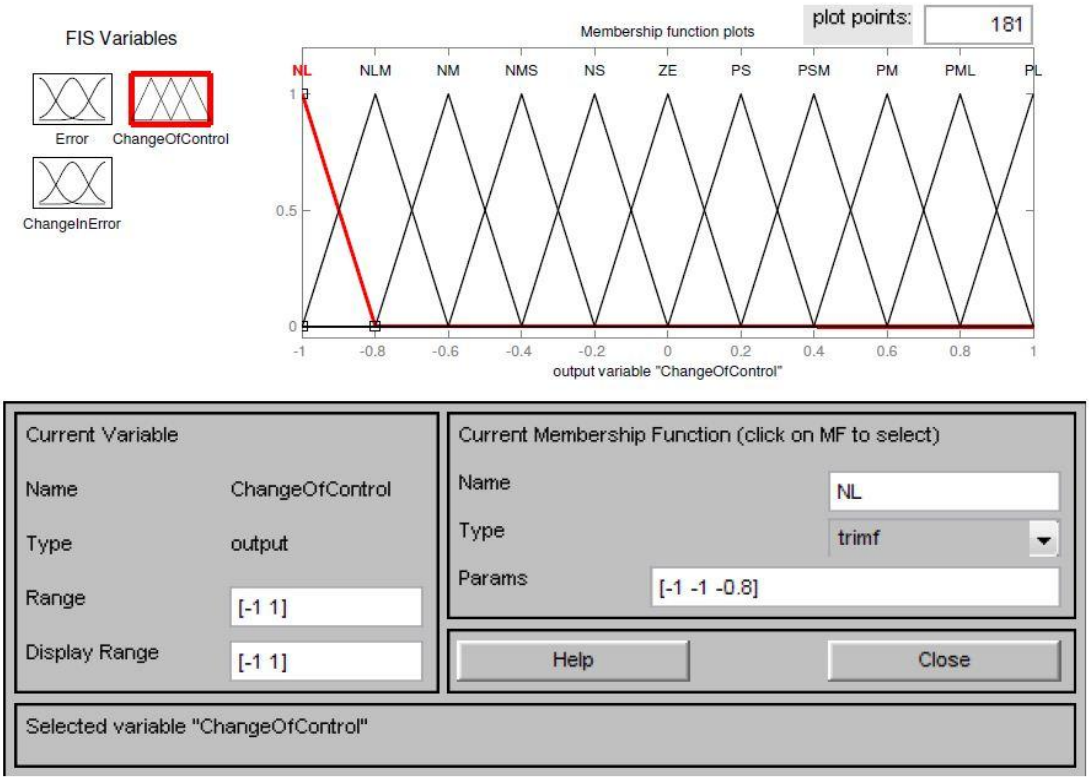


Fig 40: Modified membership function for the output Change of control (ω_{Sl})

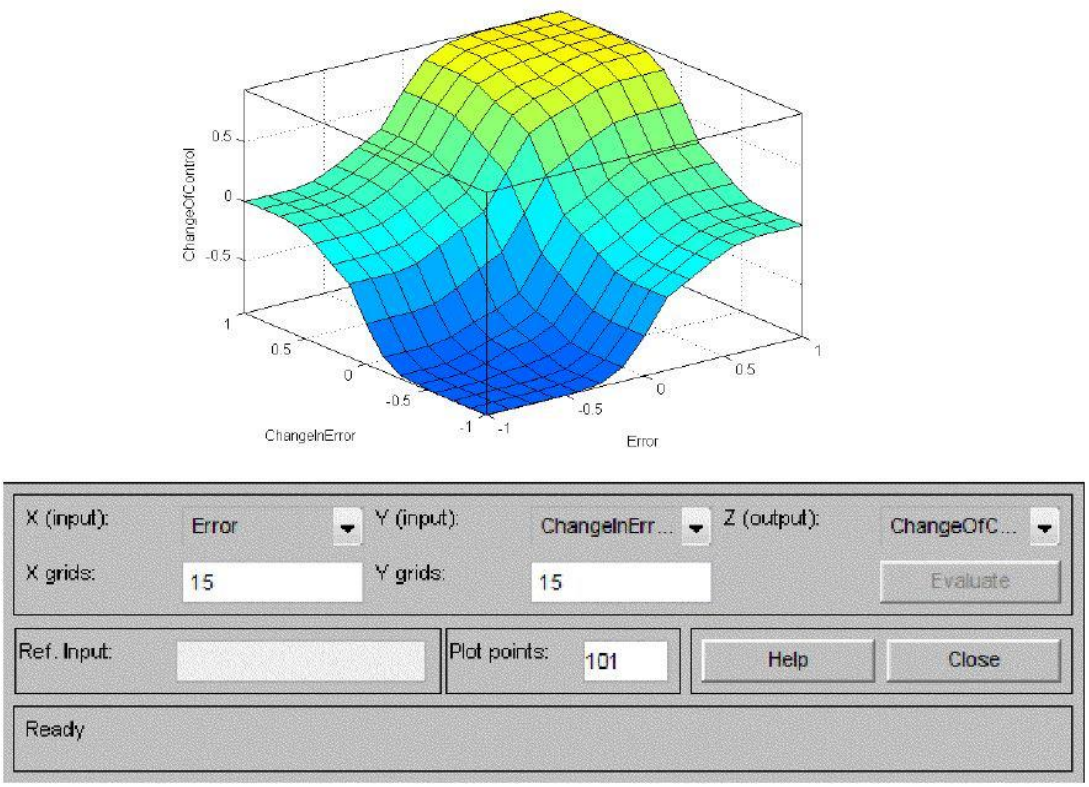


Fig 41: Three dimensional plot of the control surface for the modified controller

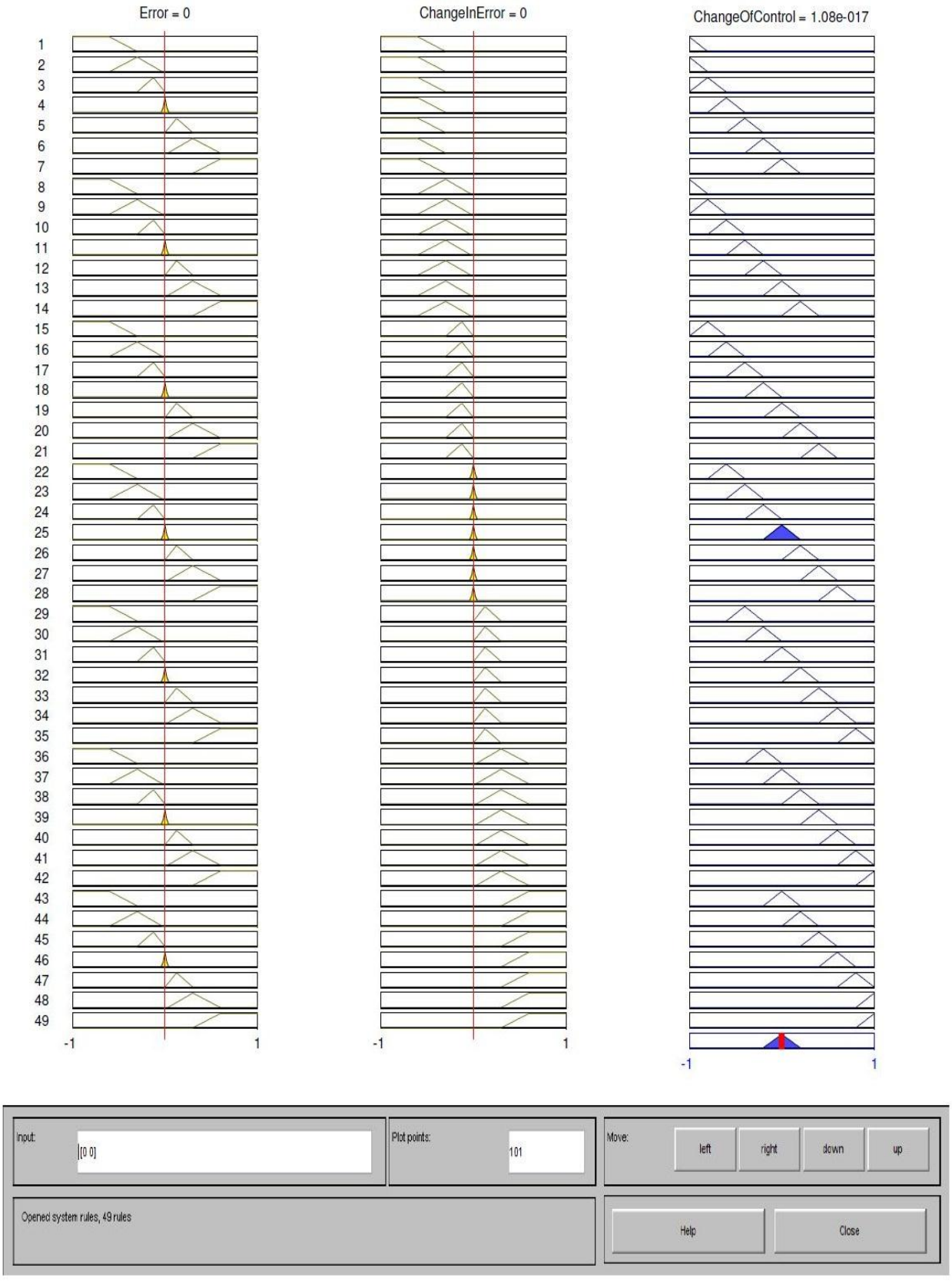


Fig 42: Rule viewer for the modified FLC with inputs **e = 0** and **Δe = 0**

Fig 43: Rule viewer for the modified FLC with inputs **e = -0.9** and **Δe = -0.3**

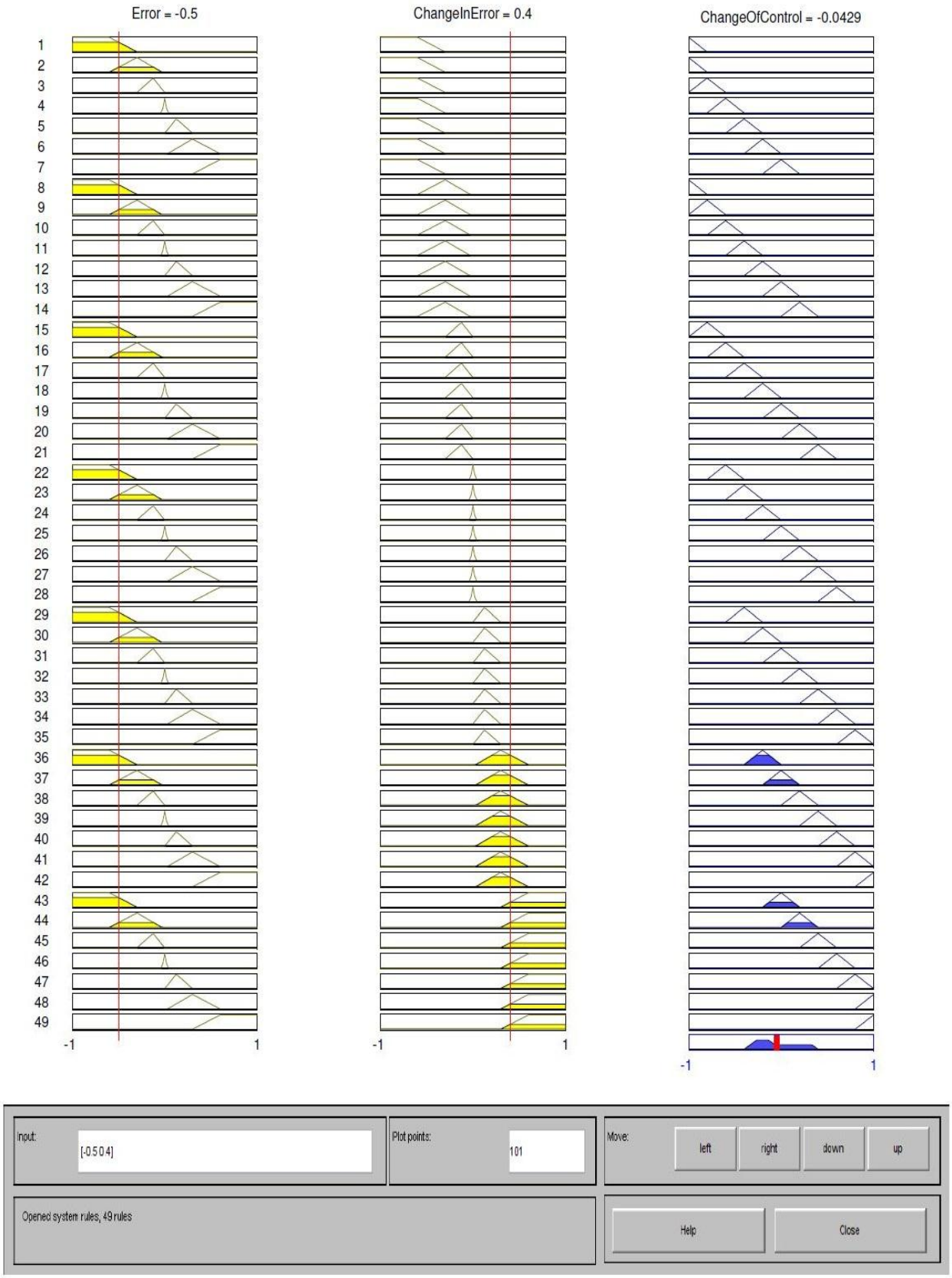


Fig 44: Rule viewer for the modified FLC with inputs **e = -0.5** and **Δe = 0.4**

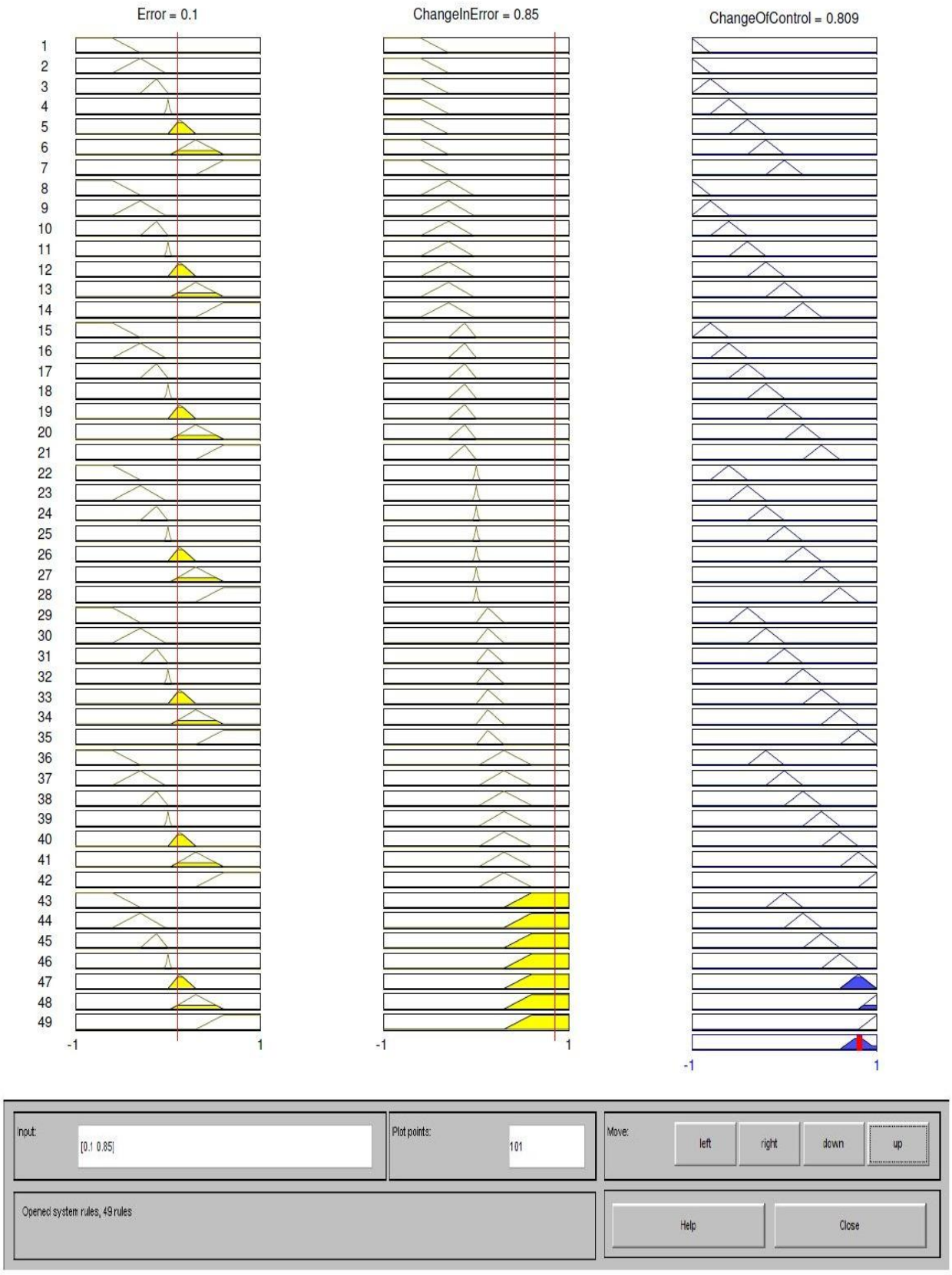


Fig 45: Rule viewer for the modified FLC with inputs **e = 0.1** and **Δe = 0.85**

7.3 MATLAB Simulations with Variations in Reference Speed and Load

Using the above modified Fuzzy Logic Controller the model given in Fig 25 was simulated. The reference speed and load torque waveforms were varied and simulations for the same were carried out. The motor speed waveforms for all the simulations were viewed and these are shown below.

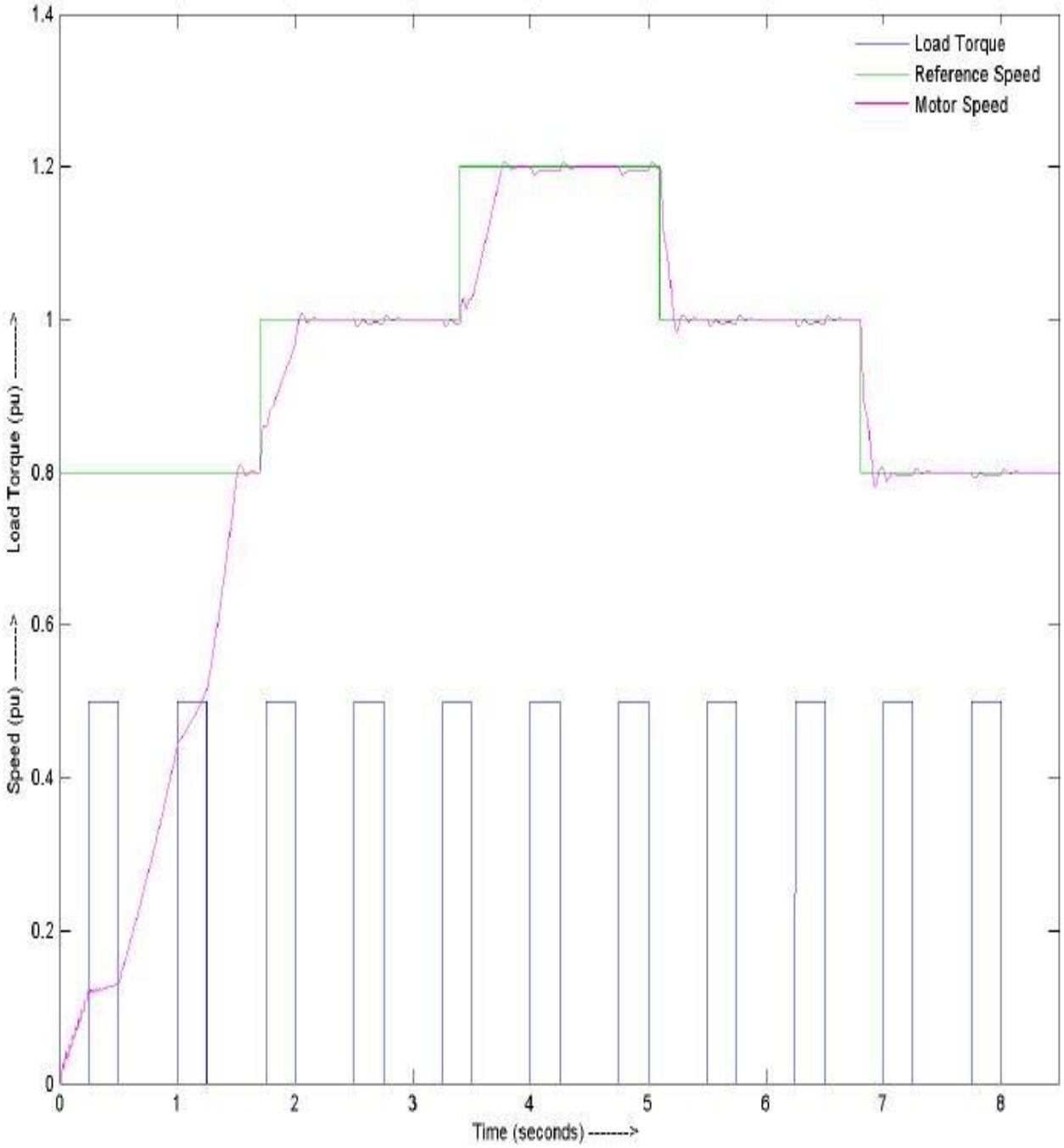


Fig 46: Simulation results with reference speed as a staircase waveform and an intermittent load

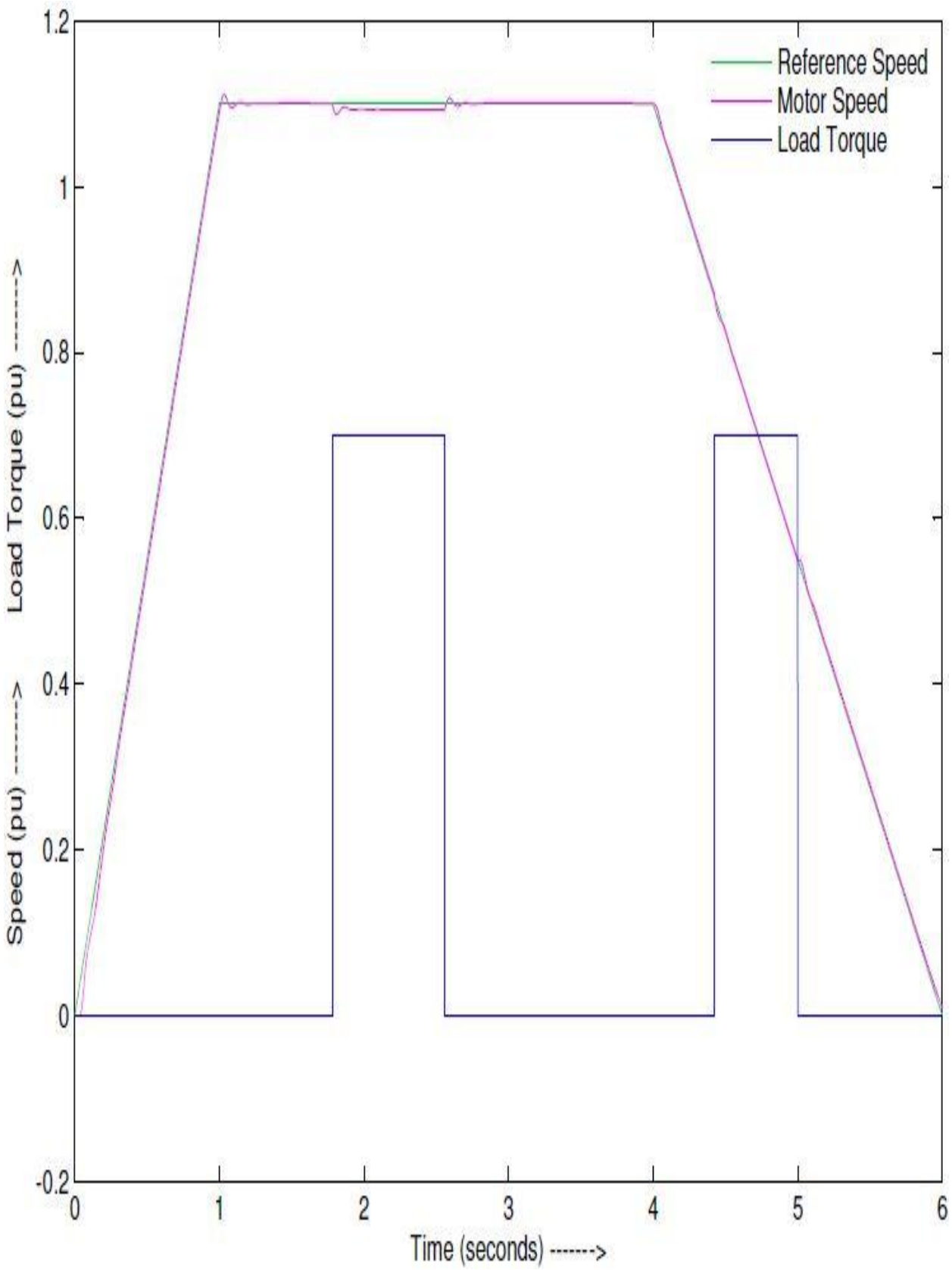


Fig 47: Simulation results with trapezoidal reference speed waveform and intermittent load

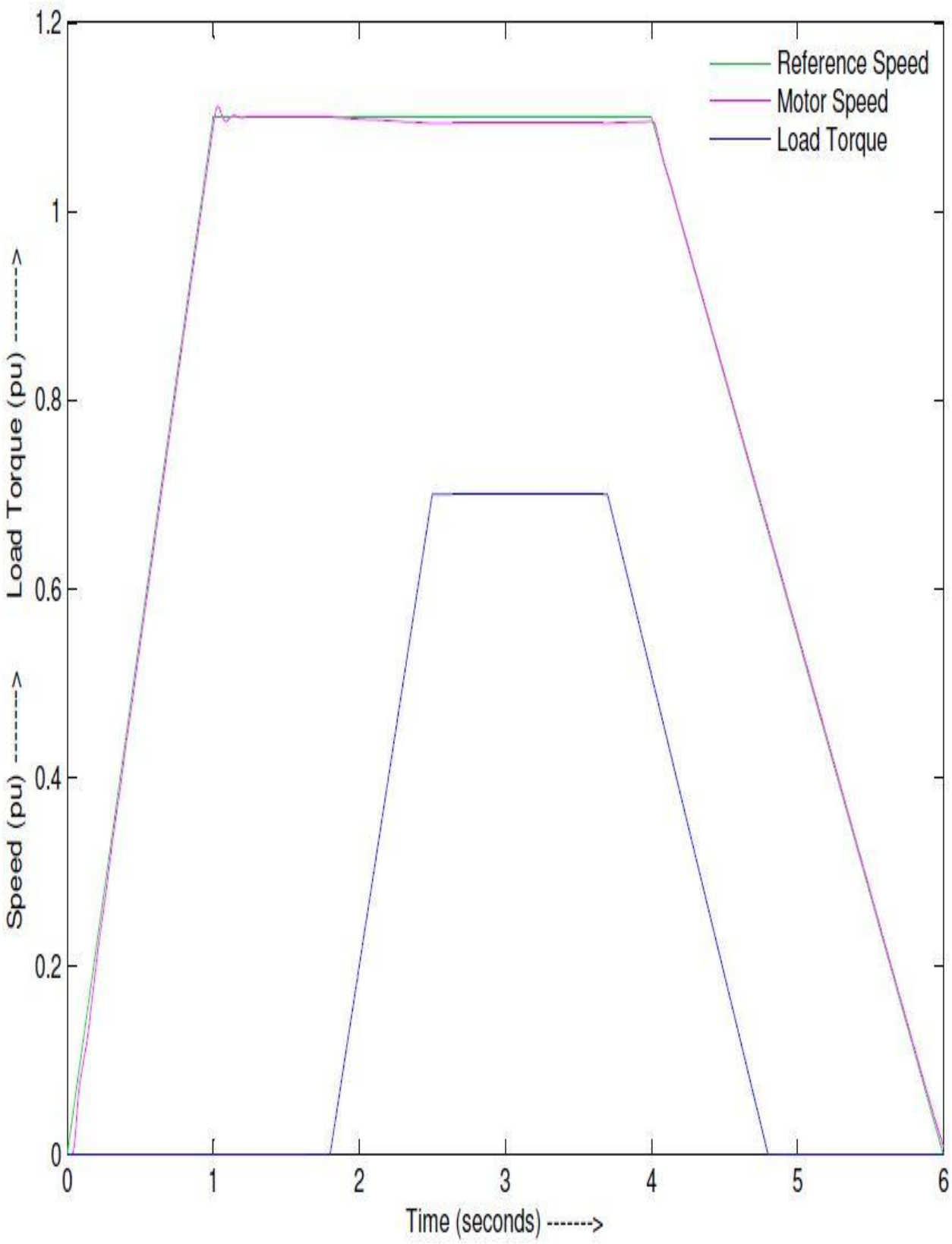


Fig 48: Simulation results with trapezoidal reference speed and load torque waveforms with load torque less than 1 p.u.

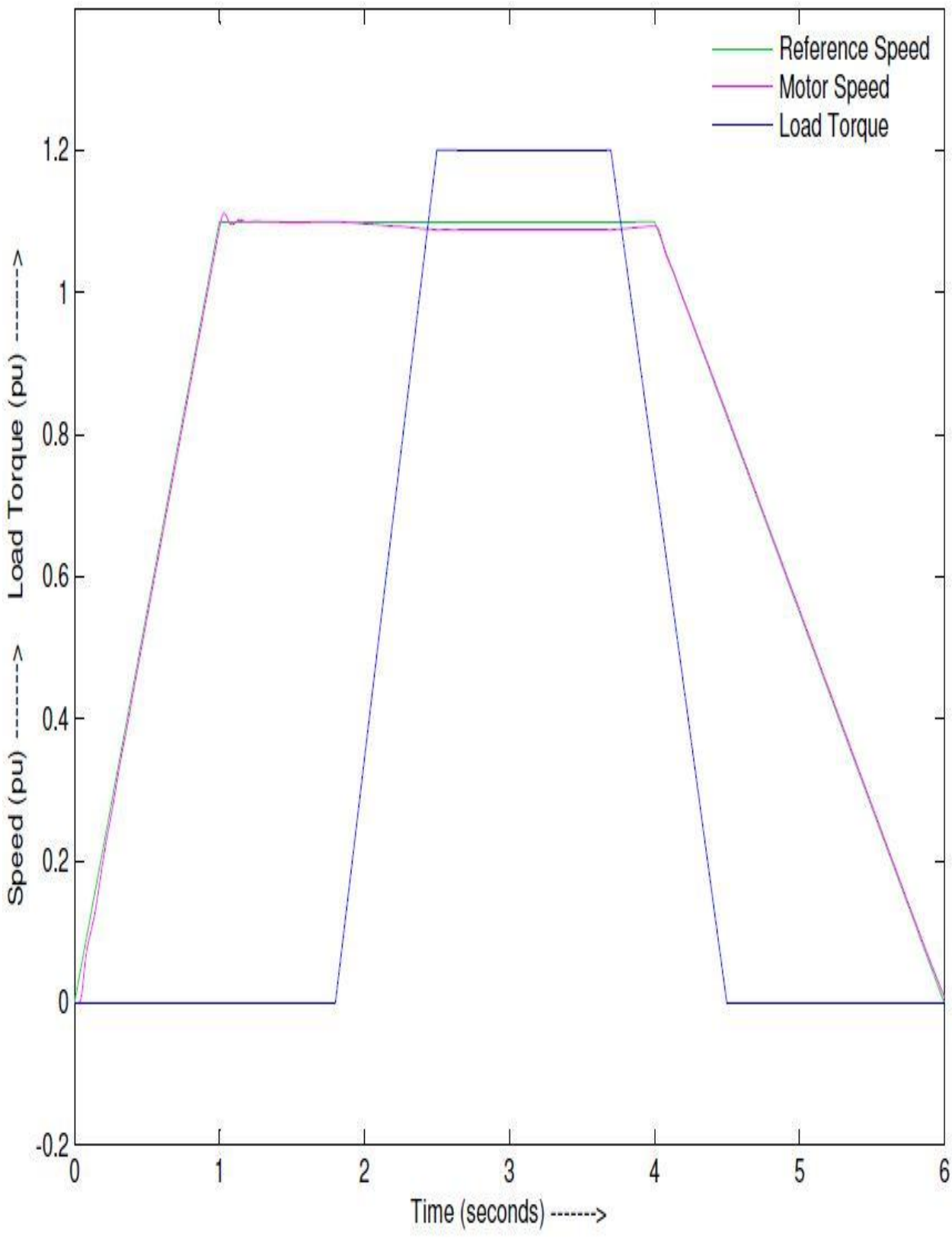


Fig 49: Simulation results with trapezoidal reference speed and load torque waveforms with load torque greater than 1 p.u.

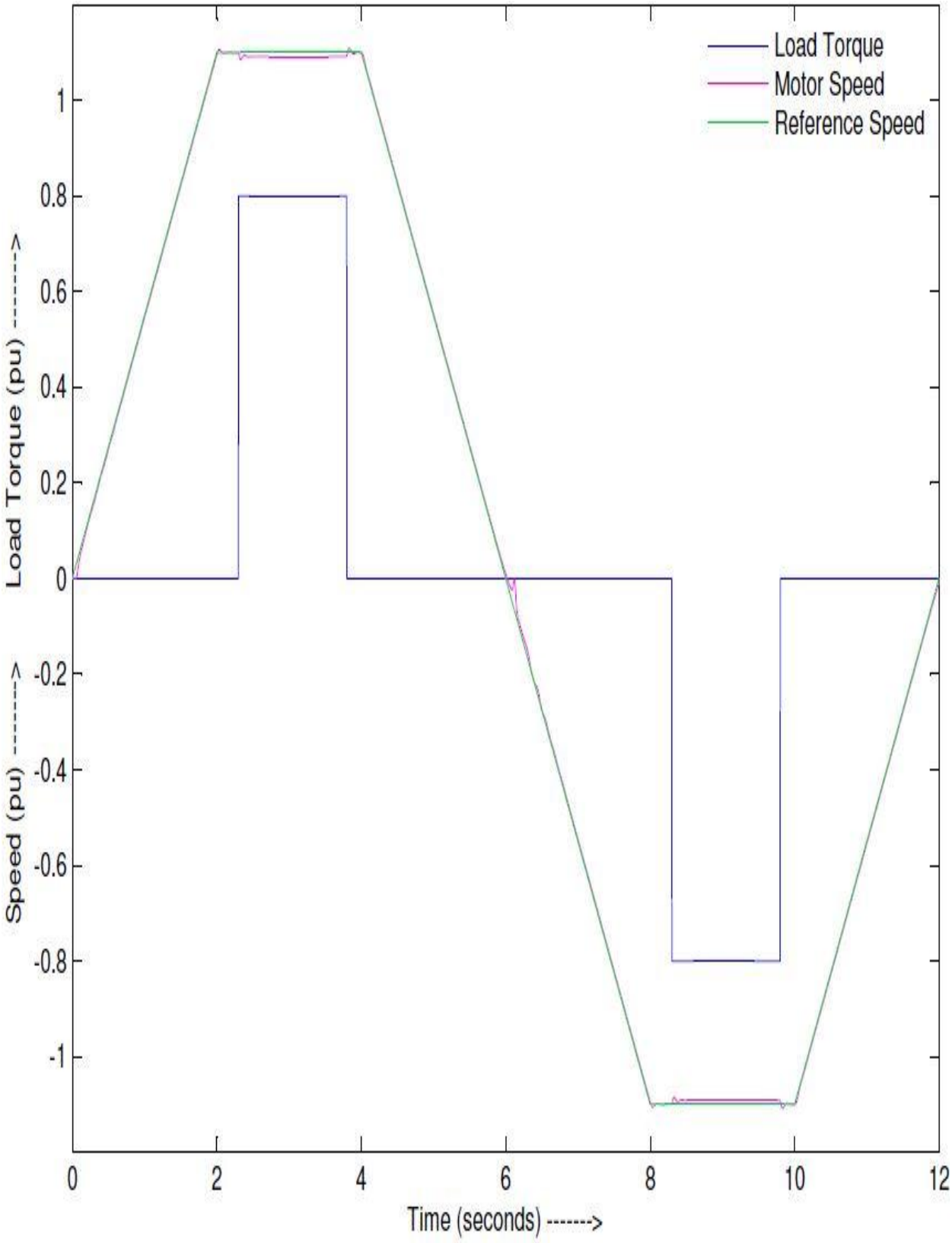


Fig 50: Simulation results with trapezoidal reference speed and rectangular load torque waveforms

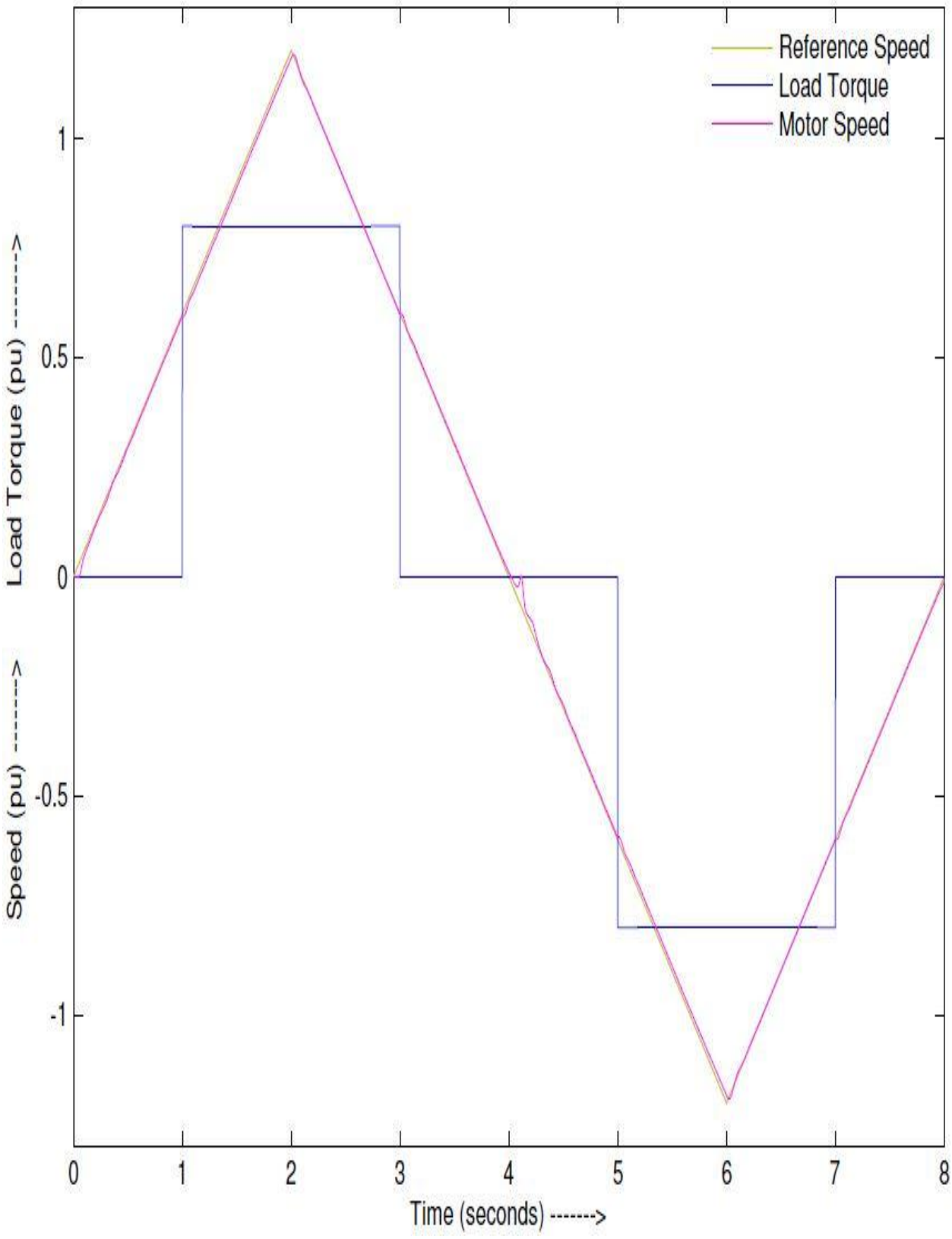


Fig 51: Simulation results with triangular reference speed and rectangular load torque waveforms

7.4 Conclusion

The modified design of the Fuzzy Logic Controller was found to have a decent performance. The steady state error was found to be zero. Whenever the induction machine was loaded the speed of the machine fell, but only to a very little extent. The rise time and the settling time of the system were not affected much, but the peak overshoot of the system was found to have reduced as compared to the earlier design. Hence, this controller can now be used in other applications. But now the system has to be optimized so as to achieve an optimum value for the rise time, settling time and peak overshoot.

CHAPTER 8

Conclusion

"To succeed, jump as quickly at opportunities as you do at conclusions"

(Benjamin Franklin)

8.1 Fuzzy Logic and Conventional Controllers: A Comparison

- The simplification or linearization of the non-linear system under consideration has to be performed by the conventional control methodologies like PI, PD and PID since their construction is based on linear system theory. Hence, these controllers do not provide any guarantee for good performance [14]. They require complex calculations for evaluating the gain coefficients. These controllers however are not recommended for higher order and complex systems as they can cause the system to become unstable. Hence, a more heuristic approach is required [12] for choice of the controller parameters which can be provided with the help of *fuzzy* logic, where we can define variables in a subjective way. Thus we can avoid the numerical complicacy involved in higher order systems.
- Fuzzy logic provides a certain level of artificial intelligence to the controllers since they try to imitate the human thought process. This facility is not available in the conventional controllers.

8.2 Discussion

So far, in this project, we have studied about the fundamentals of induction motor drives, the main topic of concern being speed control. Our focus is to develop a Fuzzy Logic Based Controller so as to achieve precision in control. The controller attempts to attain a certain level of human intelligence by utilizing the linguistic variables instead of numerical ones. Its main advantage is that it completely avoids the mathematical computations, which relieves the designer from using cumbersome techniques. All a fuzzy logic controller needs is a set of if-then rules and a knowledge base, which can be easily provided by the programmer. Hence it becomes simpler to implement fuzzy logic for the design of controllers for higher order systems.

In the project, we have designed a Fuzzy Logic Controller to be utilized in the speed control on induction motor. The designing has been done with the help of MATLAB. This controller takes in crisp inputs, viz. speed error (e) and change in error (Δe) and gives an output called change in control. The output changes according to the rules laid down by the user. These have been verified with the help of FIS rule Viewer. Four different values of e and Δe were taken and the results were obtained as shown.

After the simulation of the of the block diagram in MATLAB/SIMULINK®, it was found that the fuzzy logic controller used in the simulation worked quite effectively. The advantages of the Fuzzy Logic Controller used in the simulation were as follows:

- The overshoots in the system was very less as compared to conventional PI controller.
- The settling time was less.
- The speed tended to approach the reference speed even when it was higher than the base speed or very low as compared to the same, unlike the PI Controller.
- The designing of the control mechanism was not very cumbersome.

The disadvantages of the Fuzzy Logic Controller used were:

- The rise time was little higher as compared to the conventional PI controller.
- After the change in reference speed from base speed, the actual speed did not exactly follow it, but was found to be almost equal to it.

The Fuzzy Controller was then tuned and the some simulations were run. It was found that now the motor speed exactly follows the reference speed even after the speed changes. The modified Fuzzy Logic Controller also works fine when the reference speed is a ramp function. Hence, this modified controller is superior to that of the prior controller.

8.3 Future Scope

Simulation of the block diagram for speed control of induction motor given in Fig 25 has been performed in MATLAB/SIMULINK® and the results have been studied. The Fuzzy Logic Controller was designed and tuned so as to achieve desirable results. This controller can be implemented in different practical applications of induction motors, the feasibility of the controller in the corresponding applications can be studied and changes can be made according to the requirement. Different strategies like Genetic Algorithm can also be applied for tuning the controller. Also, instead of just fuzzy controller, a neuro-fuzzy controller can be developed based on this thesis.

References

[1] J. G. Ziegler and N. B. Nichols, "Optimum settings for automatic controllers," *Trans. ASME*, vol. 64, pp. 759–768, 1942.

[2] G. Mallesham and A. Rajani, "Automatic Tuning of PID Controller using Fuzzy Logic," 8th International Conference on Development and Application Systems, Suceava, Romania, pp. 120 – 126, 2006.

[3] L. A. Zadeh, "Fuzzy sets," Information and Control, vol. 8, pp. 338-353, 1965.

[4] G. El-Saady, A.M. Sharaf, A. Makky, M.K. Sherriny, and G. Mohamed, "A High Performance Induction Motor System Using Fuzzy Logic Controller," IEEE Trans. 0-7803-1772-6/94, pp. 1058-1061, 1994.

[5] Pavol Fedor and Daniela Perduková, "A Simple Fuzzy Controller Structure," Acta Electrotechnica et Informatica No. 4, Vol. 5, pp. 1-4, 2005.

[6] Ramón C. Oros, Guillermo O. Forte, Luis Canali, "Scalar Speed Control of a d-q Induction Motor Model Using Fuzzy Logic Controller", Departamento de Electrónica, Facultad Regional Córdoba, Universidad Tecnológica Nacional, Conf. paper.

[7] R.Ouiguini, K. Djeffal, A.Oussedik and R. Megartsi, "Speed Control of an Induction Motor using the Fuzzy logic approach.", ISIE'97 - Guimariies, Portugal, IEEE Catalog Number: 97TH8280, vol.3, pg. 1168 – 1172.

[8] Yau-Tze Kao and Chang-Huan Liu, "Analysis and Design of Microprocessor-Based Vector-Controlled Induction Motor Drives," IEEE Transactions on Industrial Electronics, Vol. 39, pp. 46 – 54, 1 February, 1992.

[9] M.Chow, A. Menozzi and F. Holcomb ,"On the Comparison of Emerging and Conventional Techniques for DC Motor Control," proc. IECON , pp. 1008-1013, 1992.

[10] Abdullah I. Al-Odienat, Ayman A. Al-Lawama, "The Advantages of PID Fuzzy Controllers Over The Conventional Types," American Journal of Applied Sciences 5 (6): 653-658, 2008, ISSN 1546-9239, pp. 653 – 658.

[11] Gopal K. Dubey, "Fundamentals of Electrical Drives", Narosa Publishing House Pvt. Ltd., 2001, chap. 6.

[12] J. Martínez García, J.A. Domínguez, "Comparison between Fuzzy logic and PI controls in a Speed scalar control of an induction machine," CIRCE – ge3 – Departamento de Ingeniería EléctricaC.P.S., Universidad de Zaragoza, Conf. Paper

[13] D. P. Kothari, I. J. Nagrath, "Electric Machines", Tata McGraw – Hill Education Private Limited, ISBN-13: 978-0-07-058377-1, ISBN-10: 0-07-058377-3, 2004, chap. 9.

[14] J.-S. R. Jang, C.-T. Sun, E. Mizutani, "Neuro-Fuzzy and Soft Computing," Pearson Education Pte. Ltd., ISBN 81-297-0324-6, 1997, chap. 2, chap. 3, chap. 4.

[15] I. J. Nagrath, M. Gopal, "Control Systems Engineering", New Age International Publishers, 2007, chap. 3.

[16] MATLAB/SIMULINK® version 2009a, The MathWorks Inc., USA

[17] Fuzzy Inference Systems reference manual for MATLAB/SIMULINK® version 2009a