Chapter One
Introduction

1.1 Background

For a large variety of Industrial applications dc motors are being used such as elevators, electric trains, hoists, heavy metal rolling mills, automobiles, robots, hand power tools and food blenders. Micro-machines are electric machines with parts the size of red blood cells, and find many applications in medicine.

Several conventional and numeric controller types, the controllers can be: Proportional integral (PI), proportional integral derivative (PID), Fuzzy Logic Controller (FLC) or the combination between them: Fuzzy-Neural Networks, Fuzzy-Genetic Algorithm, Fuzzy-Ants Colony, Fuzzy-PID.

The proportional – integral – derivative (PID) controller operates the majority of the control system in the world. PID controllers provide robust and reliable performance for most systems if the PID parameters are tuned properly.

DC drives are simplicity, ease of application, high reliabilities, flexibilities and favorable cost have long been a backbone of industrial applications.

The advancements in new techniques and a greater understanding of existing capabilities enable the PID to not only improve loop performance but to individually optimize unit operations.

1.2 Problem Statement

High performance motor drives are very important in industrial as well as other purpose applications, a high-performance motor drive system must have good dynamic speed command tracking and load regulating response to perform task, PID controller can be used to minimize speed error in DC motor, PID controller need to adjust the three gain parameters during system run.

1.3 Proposed Solution

Design of self-tuning PID controller using fuzzy logic for DC motor speed, use fuzzy logic controllers to improve parameter of PID controller ability to handle disturbances.

1.4 Objectives

- To model a Separately excited DC motor
- To control the DC motor speed with conventional controlling (PID) methods
- To control the DC motor speed with FUZZY-PID controller
- To analyze the sensitivity, evaluate and compare the effects of different types of MFs in the Fuzzy PID DC motor speed control.
- Compare the different speed controlling techniques.

1.5 Methodology

Implementation of self-tuned PID controller (Fuzzy-PID) for speed control of DC motor based on MATLAB. The basic property of DC motor is that speed of DC motor can be adjusted by varying its terminal voltage. It can easily be deduced that Fuzzy-PID can maintain the speed at desired values irrespective of change of load.

The self-tuning PID-type fuzzy controller is an auto-adaptive PID controller that is designed by using an incremental fuzzy logic controller in place of the proportional term in the conventional PID controller to tune the parameters of PID controller on line by fuzzy control rules.

The advantage of fuzzy logic for online tuning of PID controller has been used to obtain robust speed control of a dc motor with an accurately modeled.

The model and simulation methods for analyzing, testing and developing of dc motor using MATLAB/SIMULINK and PID parameters were obtained by using Fuzzy sets.

1.6 Research layout

- ***Chapter One***: Introduced the general overview, the problems that solved by it and the objectives that will going to be achieved.
- ***Chapter Two***: Is about control system, PID controller, fuzzy logic control system and electric motors.
- ***Chapter Three***: Is about simulation of speed control system with conventional PID controller, Fuzzy controller and Fuzzy-PID self-tuning controller using MATLAB/SIMULINK.
- ***Chapter Four***: Is about simulation results, it shows the comparison between controllers.
- ***Chapter Five***: Conclusion and Recommendations.

Chapter Two

Literature Review

2.1 Overview

A basic control system has an *input*, a *process*, and an *output*. The basic objective of control system is of regulating the value of some physical variable or causing that variable to change in a prescribed manner in time.

Control systems are typically classified as *open loop* or *closed-loop*. *Open-loop control systems* do not monitor or correct the output for disturbances whereas *closed-loop control systems* do monitor the output and compare it with the input. In a closed-loop control system if an error is detected, the system corrects the output and there by corrects the effects of disturbances. In closed-loop control systems, the system uses *feedback*, which is the process of measuring a control variable and returning the output to influence the value of the variable. [1]

Block diagrams display the operational units of a control system. Each block in a *component block diagram* represent some major component of the control system, such as measurement, compensation, error detection, and the plant itself. It also depicts the major directions of information and energy flow from one component to another in a control system.

A block can represent the component or process to be controlled. Each block of a control system has a transfer function (represented by differential equations) and defines the block output as a function of the input.

Control system design and analysis objectives include: producing the response to a transient disturbance follows a specified pattern (over-damped or under damped), minimizing the steady-state errors, and achieving the stability. [1]

2.2 Conventional Control System

A basic controls system is shown in Figure (2.1). The process ***(p)*** or plant is the object to the controlled. It is inputs are ***u(t)***, it is outputs are ***y(t)***, and reference inputs is ***r(t)***. In this section provide an overview of the steps taken to design the controller shown in Figure (2.1). Basically, there are modeling, controllers design, and performance evaluation.

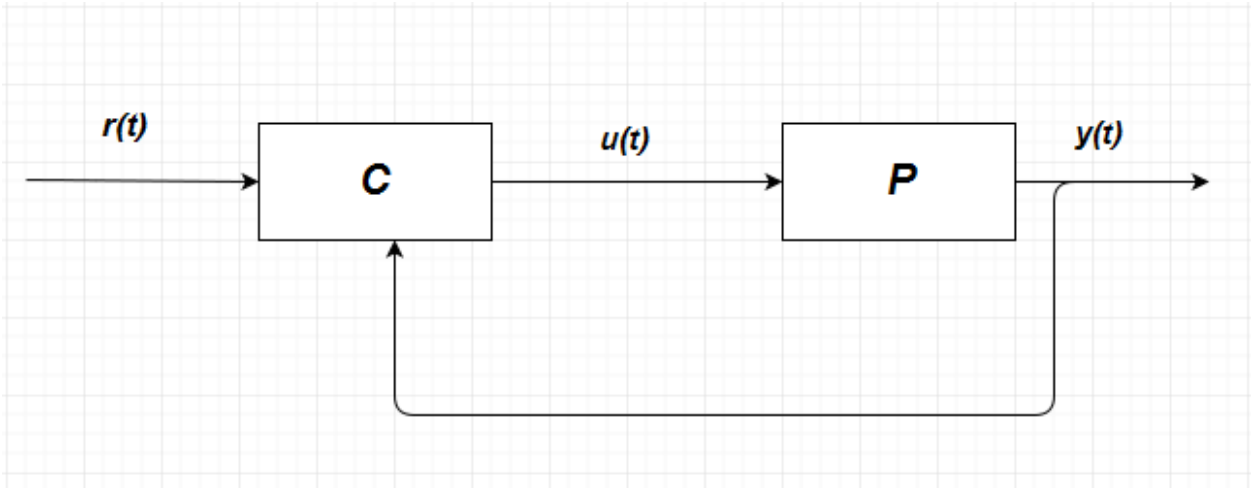


Figure (2.1) control system

2.3 Performance Specifications

Control systems are designed to perform specific tasks. The specifications of a control system must be given before the design process begins. The specifications may be given in terms of transient response requirements such as: [1]

- ***Overshoot:*** The amount by which the system output response proceeds beyond the desired response.
- ***Peak Time*****:** The peak time is the time required for the response to reach the first peak of the overshoot.
- ***Peak Value*****:** The maximum value of the output, reached after application of the unit step input after time.
- ***Rise Time*****:** time required for the response to a unit step input to rise from 10 to 90% of its final value. For underdamped second-order system, the 0% to 100% rise time is normally used. For overdamped systems, the 10% to 90% rise time is common.
- ***Settling Time*****:** The time required for the system output to settle within a certain percentage of the input amplitude.
- ***Stability*****:** That characteristic of a system defined by a natural response that decays to zero as time approaches infinity.
- ***Steady-State Error*****:** The difference between the input and output of a system after the natural response has decayed to zero.

2.4 PID Controller

PID controller is a three-term controller that has a long history in the automatic control field, starting from the beginning of the last century (bennett,2000). PID controller is name commonly given to three-term control. The mnemonic PID refers to the first letters of the names of the individual terms that make up the standard three-term controller. These are *P* for the proportional term, I for the integral term and *D* for the derivative term in the controller. Three term or PID controllers are Probably the most widely used industrial controller. [2]

PID controller has several important functions: it provides feedback, it has the ability to eliminate steady state offsets through integral action, it can anticipate the future through derivative action.

PID controllers are sufficient for many control Problems, particularly when Process dynamics are benign and the Performance requirement are modest. PID controllers come in many different forms. There are standalone systems in boxes for one or few loops, which are manufactured by the hundred thousand yearly. PID control is an import part of a distributed control system. The controllers are also embedded in many special purpose control systems. In process control, more than 95% of the control loops are of PID type, most loops are actually PI control.

PID control is often combined with logic, sequential machines, selectors, and simple function blocks to build the complicated automation systems used for energy production, transportation, and manufacturing. Many sophisticated control strategies, such as model predictive control, are also organized hierarchically.

PID controllers have survived many changes in technology ranging from Pneumatics to microprocessor via electronic tubes, transistor, integrated circuits. The microprocessor has had a dramatic influence on the PID controller. Practically all PID controller made today are based on microprocessor. This has given opportunities to provide additional features like automatic tuning, gain scheduling, and continuous adaptation. [2]

2.4.1 PID Structure

The structure of PID include proportional term, integral term and derivative term. The PID controller is mainly to adjust appropriate proportional gain(k_P), integral gain (k_i), and differential gain (k_d) to achieve the optimal control performance. PID structure as shown in figure (2.2), r(t) is reference, *e(t)* is error, *u(t)*is controller output and *y(t)* system output.

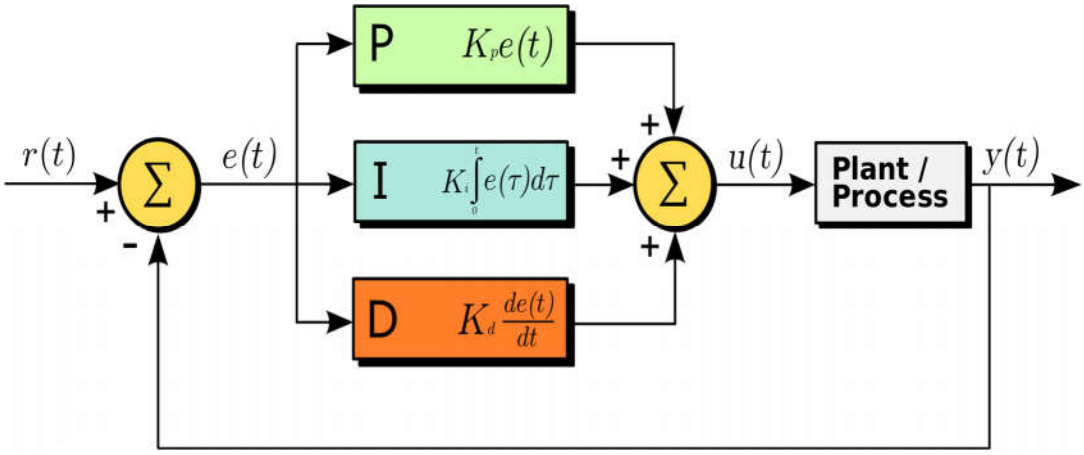


Figure (2.2) PID Controller Structure

Relation between the input *e(t)* and output *u(t)* can be formulated in the following:

$$e(t)=r(t)-y(t) \tag{2.1}$$

$$U(t) = k_P.e(t)+k_I.\int_0^t e(t)dt+k_D.\frac{de(t)}{dt} \tag{2.2}$$

The transfer function is expressed (Laplace domain):

$$C(S) = \frac{U(S)}{E(S)} = k_P + \frac{ki}{S} + k_D S \tag{2.3}$$

2.4.1.i Proportional Term

The proportional term produces an output value that is proportional to the current error value. The proportional response can be adjusted by multiplying the error by a constant K_p, called the proportional gain constant.

The proportional term is given by:

$$P_{out} = k_P\, e(t) \tag{2.4}$$

A high proportional gain results in a large change in the output for a given change in the error. If the proportional gain is too high, the system can become unstable. A small gain results in a small output response to a large input error, and a less responsive or less sensitive controller.

If the proportional gain is too low, the control action may be too small when responding to system disturbances. [3]

2.4.1.ii Integral Term

The contribution from the integral term is proportional to both the magnitude of the error and the duration of the error. The integral in a PID controller is the sum of the instantaneous error over time and gives the accumulated offset that should have been corrected previously.

The accumulated error is then multiplied by the integral gain (*Ki*) and added to the controller output.

The integral term is given by:

$$I_{out} = k_i \int_0^t e(t)\,dt \tag{2.5}$$

The integral term accelerates the movement of the process towards set point and eliminates the residual steady-state error that occurs with a pure

proportional controller. However, since the integral term responds to accumulated errors from the past, it can cause the present value to overshoot the set point value. [3]

2.4.1.iii Derivative Term

The derivative of the process error is calculated by determining the slope of the error over time and multiplying this rate of change by the derivative gain K_d. The magnitude of the contribution of the derivative term to the overall control action is termed the derivative gain (K_d).

The derivative term is given by:

$$\boldsymbol{D_{out} = k_d \frac{de(t)}{dt}} \tag{2.6}$$

Derivative action predicts system behavior and thus improves settling time and stability of the system. An ideal derivative is not causal, so that implementations of PID controllers include an additional low-pass filtering for the derivative term to limit the high-frequency gain and noise. [3]

2.4.2 Parallel PID Controller

PID controller in Parallel form (also known as standard form, ISA form or non-interacting form), has control equation:

$$\boldsymbol{U=K\,[1 + \frac{1}{S.Ti} + S.T_d].\,E} \tag{2.7}$$

The controller actions (P, I and D) act independently as can be seen in the corresponding block diagram representation. show figure (2.3)

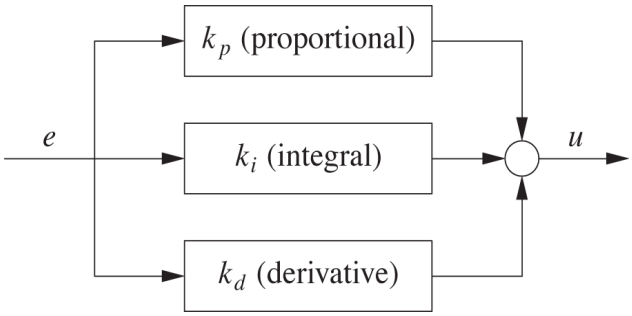


Figure (2.3) PID Controller in Parallel

2.4.3 Series PID Controller

PID controller in series form (also known as interacting form), has the control equation:

$$U = [1 + \frac{1}{STi} + ST_d\,]\,(1+S.T_d).E \qquad (2.8)$$

The controller actions (P, I and D) act dependently as can be seen in the corresponding block diagram representation. show figure (2.4)

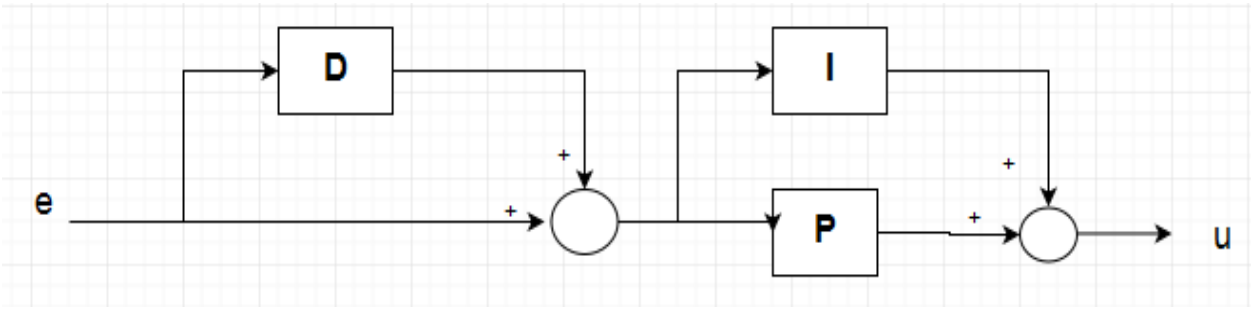


Figure (2.4) PID Controller in Series

2.4.4 Tuning PID Controller

Tuning a control loop is the adjustment of its control parameters (proportional gain, integral gain, derivative gain) to the optimum values for the desired control response. Stability is a basic requirement, but beyond that, different systems have different behavior, different applications have different requirements, and requirements may conflict with one another.

this section describes some traditional methods for loop tuning:

2.4.4.i Manual Tuning

If the system must remain online, one tuning method is to first set k_i and k_d values to zero. Increase the k_P until the output of the loop oscillates, then the k_P should be set to approximately half of that value for a "quarter amplitude decay" type response. Then increase k_i until any offset is corrected in sufficient time for the process. However, too much k_i will cause instability.

Finally, increase k_d, if required, until the loop is acceptably quick to reach its reference after a load disturbance. However, too much k_d will cause excessive response and overshoot. A fast PID loop tuning usually overshoots slightly to reach the set point more quickly. [3]

Table (2.1) Effects of Increasing Parameter Independently

Parameter	Rise time	Overshoot	Settling time	Steady-state error	Stability[14]
K_p	Decrease	Increase	Small change	Decrease	Degrade
K_i	Decrease	Increase	Increase	Eliminate	Degrade
K_d	Minor change	Decrease	Decrease	No effect in theory	Improve if K_d small

2.4.4.ii Zigler_Nichols Tuning

It is performed by setting the ***I*** (integral) and ***D*** (derivative) gains to zero. The "***P***" (proportional) gain, k_P is then increased (from zero) until it reaches the *ultimate gain* k_u, at which the output of the control loop has stable and consistent oscillations**.** k_u and the oscillation period T_u are used to set the ***P***, ***I***, and ***D*** gains depending on the type of controller used:

Table (2.2) Ziegler–Nichols Method

Control Type	K_p	K_i	K_d
P	$0.50K_u$	-	-
PI	$0.45K_u$	$0.54K_u/T_u$	-
PID	$0.60K_u$	$1.2K_u/T_u$	$3K_uT_u/40$

These gains apply to the ideal, parallel form of the PID controller. When applied to the standard PID form, the integral and derivative time parameters T_i and T_d are only dependent on the oscillation period T_u. [3]

These 3 parameters are used to establish the correction *u(t)* from the error *e(t)* via the equation:

$$u(t) = K_p \left(e(t) + \frac{1}{T_i} \int_0^t e(\tau)d\tau + T_d \frac{de(t)}{dt} \right) \quad (2.9)$$

2.4.4.iii PID Tuning Software

Most modern industrial facilities no longer tune loops using the manual calculation methods shown above. Instead, PID tuning and loop optimization software are used to ensure consistent results. These software packages will

gather the data, develop process models, and suggest optimal tuning. Some software packages can even develop tuning by gathering data from reference changes.

Mathematical PID loop tuning induces an impulse in the system, and then uses the controlled system's frequency response to design the PID loop values. In loops with response times of several minutes, mathematical loop tuning is recommended, because trial and error can take days just to find a stable set of loop values. Optimal values are harder to find.

Some digital loop controllers offer a self-tuning feature in which very small set point changes are sent to the process, allowing the controller itself to calculate optimal tuning values.

Other formulas are available to tune the loop according to different performance criteria. Many patented formulas are now embedded within PID tuning software and hardware modules.

Advances in automated PID Loop Tuning software also deliver algorithms for tuning PID Loops in a dynamic or Non-Steady State (NSS) scenario. The software will model the dynamics of a process, through a disturbance, and calculate PID control parameters in response. [3]

2.5 Fuzzy Logic

Fuzzy logic is an extension of Boolean logic by Lotfi Zadeh in 1965 based on the mathematical theory of fuzzy sets, which is a generalization of the classicalsettheory.

By introducing the notion of degree in the verification of a condition, thus enabling a condition to be in a state other than true or false, fuzzy logic provides a very valuable flexibility for reasoning, which makes it possible to take into account inaccuracies and uncertainties.one advantage of fuzzy logic in order to formalize human reasoning is that the rules are set in natural language.[4]

Fuzzy logic is determined as a set of mathematical principles for knowledge

representation based on degrees of membership rather than on crisp membership of classical binary logic. Unlike two-valued Boolean logic, fuzzy logic is multi-valued.

It deals with degrees of membership and degrees of truth. Fuzzy logic uses the continuum of logical values between 0 (completely false) and 1 (completely true). Instead of just black and white, it employs the spectrum of colors, accepting that things can be partly true and partly false at the same time. Classical binary logic now can be considered as a special case of multi-valued fuzzy logic. [5]

2.5.1 Fuzzy Sets

The concept of a set is fundamental to mathematics. Let X be a classical (crisp) set and x an element. Then the element x either belongs to X (x ∈X) or does not belong to X (x X). That is, classical set theory imposes a sharp boundary on this set and gives each member of the set the value of 1, and all members that are not within the set a value of 0. This is known as the principle of dichotomy.

Crisp set theory is governed by a logic that uses one of only two values: true or false. This logic cannot represent vague concepts, and therefore fails to give the answers on the paradoxes.

The basic idea of the fuzzy set theory is that an element belongs to a fuzzy set with a certain degree of membership. Thus, a proposition is not either true or false, but may be partly true (or partly false) to any degree. This degree is usually taken as a real number in the interval [0,1].

A fuzzy set can be simply defined as a set with fuzzy boundaries. In the fuzzy theory, fuzzy set A of universe X is defined by function $\mu_x(x)$ called the membership function of set A. Fuzzy and crisp sets can be also presented as shown in Figure (2.5).

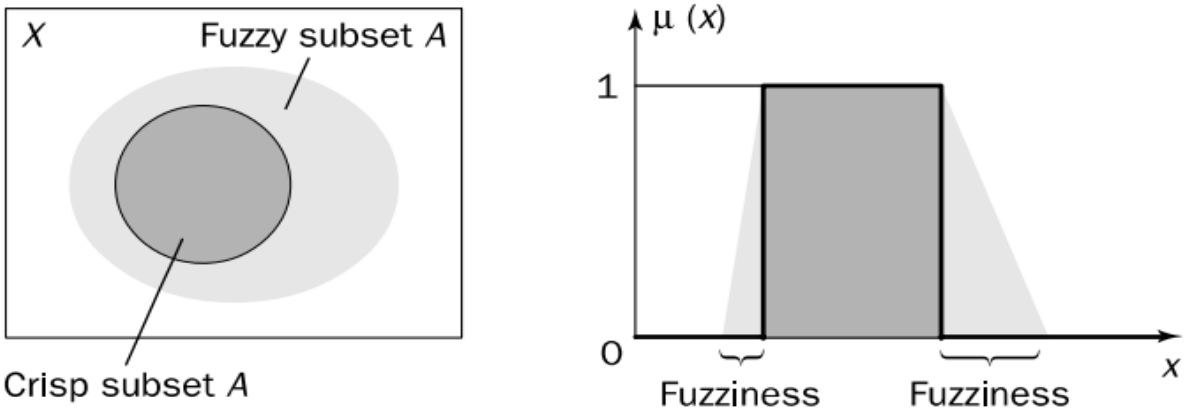


Figure (2.5) Representation of Crisp and Fuzzy Subset of X

2.5.2 Membership Function

A *membership function* (MF) is a curve that defines how each point in the input space is mapped to a membership value (degree of membership) between 0 and 1. The input space is sometimes referred to as the *universe of discourse*.

The function itself can be an arbitrary curve the shape can define as a function that suits from the point of view of simplicity, convenience, speed, and efficiency. [5]

To determine the MFs Use the knowledge of human experts or Data collected from various sensors. Among them the most popular shapes are triangular and trapezoidal because these shapes are easy to represent designer's idea and require low computation time. Show different shapes in figure (2.6)

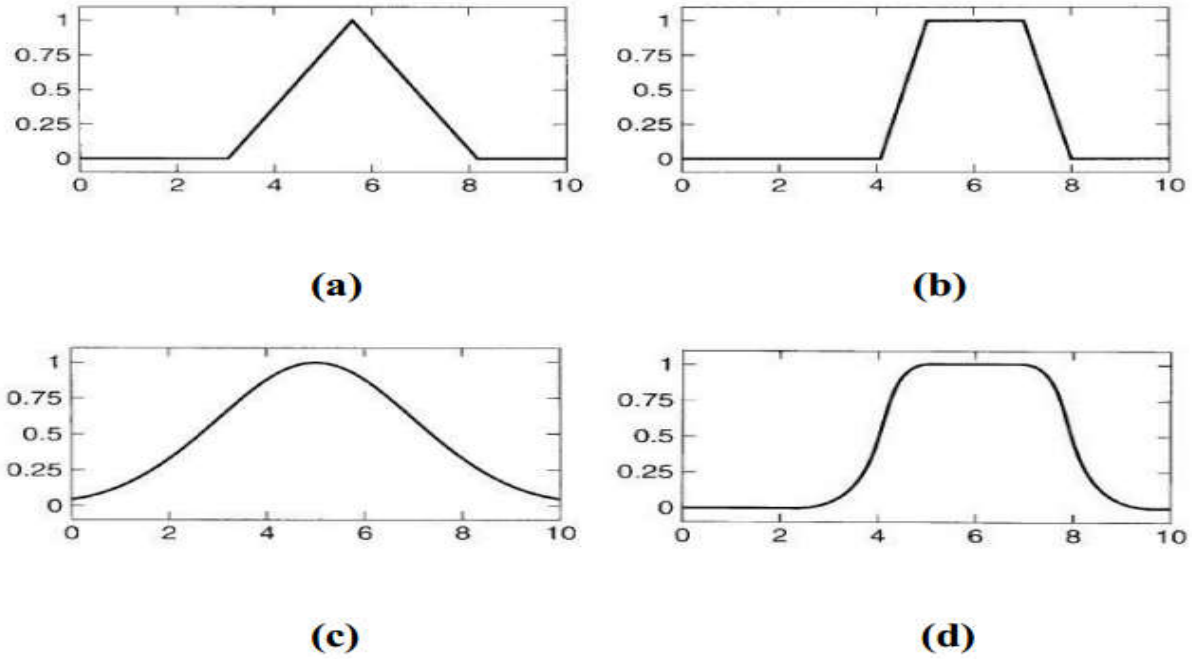


Figure (2.6) Different Types of Membership Functions (a) Triangular (b) Trapezoidal (c) Gaussian (d) Generalized Bell

2.5.3 Linguistic Variables

The root of fuzzy set theory lies the idea of linguistic variables. A linguistic variable is a fuzzy variable. The range of possible values of a linguistic variable represents the universe of discourse of that variable. The universe of discourse of the linguistic variable speed include such fuzzy subsets as very slow, slow,

medium, fast, and very fast. Each fuzzy subset also represents a linguistic value of the corresponding linguistic variable.

A linguistic variable carries with it the concept of fuzzy set qualifiers, called hedges. Hedges are terms that modify the shape of fuzzy sets. They include adverbs such as very, somewhat, quiet, more or less and slightly. Hedges can modify verbs, adjectives, adverbs or even whole sentences. They are used as:

- All-purpose modifiers, such as very, quite or extremely.
- Truth-values, such as quite true or mostly false.
- Probabilities, such as likely or not very likely.
- Quantifiers, such as most, several or few.
- Possibilities, such as almost impossible or quite possible.

2.5.4 Operations on Fuzzy Sets

The classical set theory developed in the late 19th century by Georg Cantor describes how crisp sets can interact. These interactions are called operations.

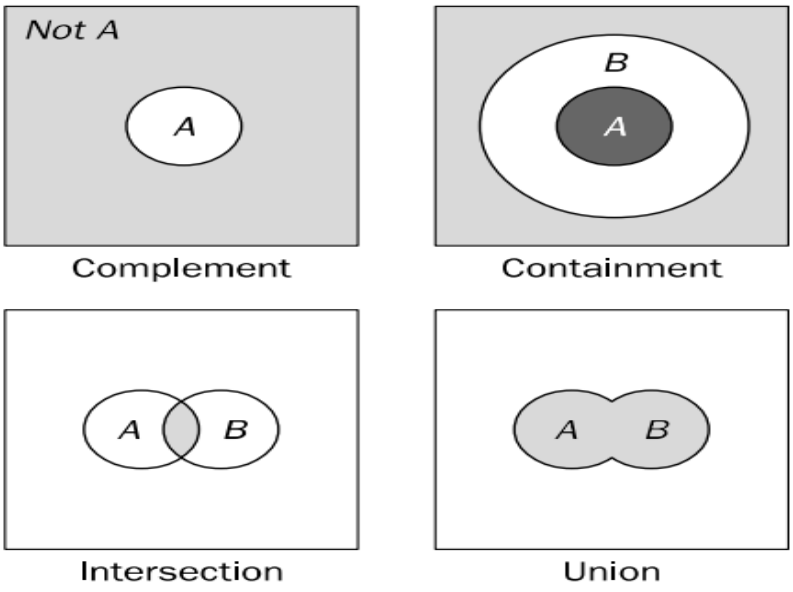


Figure (2.7) Operations on Classical Sets

- ***Complement***

The complement of a set is an opposite of this set. The set of tall men, its complement is the set of NOT tall men. When remove, the tall men set from the universe of discourse, obtain the complement. If A is the fuzzy set, its complement: -A can be found as follows:

$$\mu_{-A}(x)=1-\mu_A(x) \qquad (2.10)$$

- ***Containment***

A set can contain other sets. The smaller set is called the subset. The set of tall men contains all tall men. Therefore, very tall men are a subset of tall men. However, the tall men set is just a subset of the set of men. In crisp sets, all elements of a subset entirely belong to a larger set and their membership values are equal to 1.

In fuzzy sets, however, each element can belong less to the subset than to the larger set. Elements of the fuzzy subset have smaller memberships in it than in the larger set.

- ***Intersection***

Intersection between two sets contains the elements shared by these sets. In fuzzy sets, however, an element may partly belong to both sets with different memberships. Thus, a fuzzy intersection is the lower membership in both sets of each element.

The fuzzy operation for creating the intersection of two fuzzy sets A and B on universe of discourse X can be obtained as:

$$\mu_{A\cap B}(x)=min[\mu_A(x), \mu_B(x)] = \mu_A(x)\cap \mu_B(x), \text{ where } x\in X \qquad (2.11)$$

- ***Union***

The union of two crisp sets consists of every element that falls into either set. The union of tall men and fat men contains all men who are tall OR fat. In fuzzy sets, the union is the reverse of the intersection. The union is the largest membership value of the element in either set. The fuzzy operation for forming the union of two fuzzy sets A and B on universe X can be given as:

$$\mu_{AUB}(x)=\max[\mu_A(x), \mu_B(x)] = \mu_A(x) \cup \mu_B(x), \text{ where } x \in X \qquad (2.12)$$

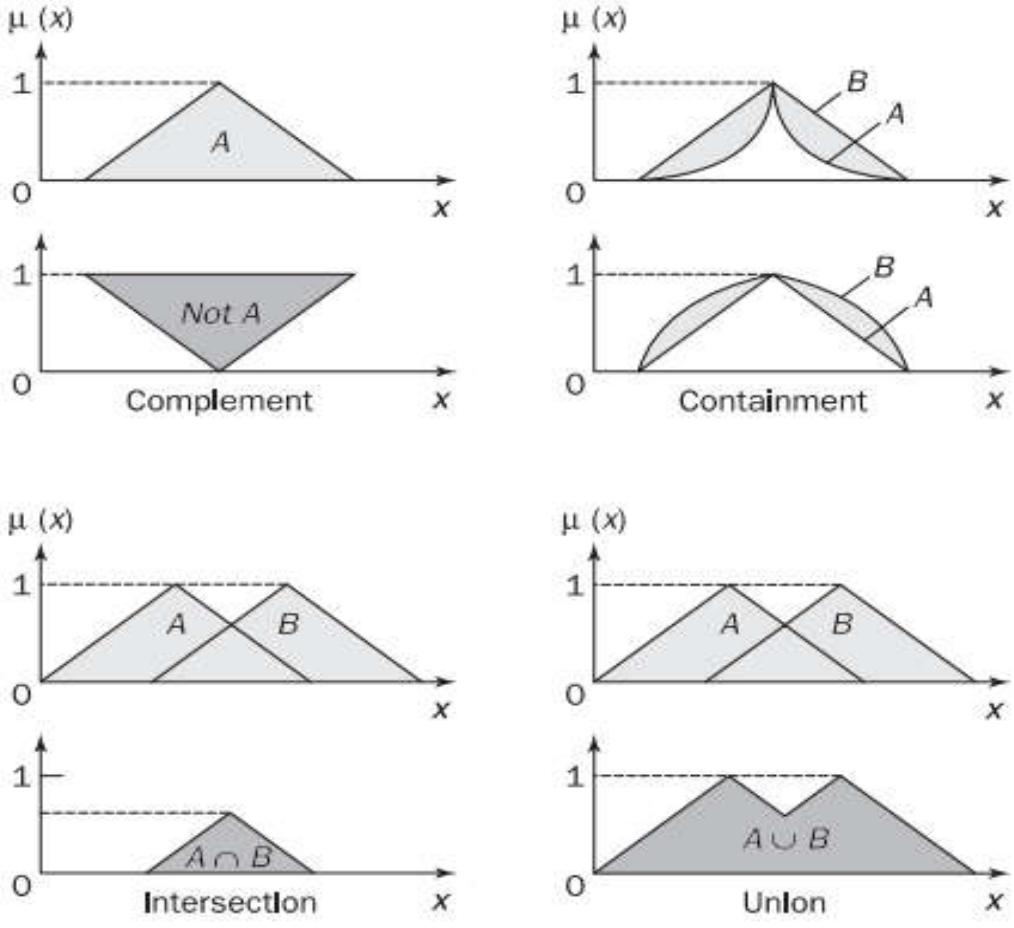


Figure (2.8) Operations of Fuzzy Sets

2.5.5 Fuzzy Rules

Fuzzy sets and fuzzy operators are the subjects and verbs of fuzzy logic. These if-then rule statements are used to formulate the conditional statements that comprise fuzzy logic.

A single fuzzy if-then rule assumes the form:

If x is A, then y is B

where *A* and *B* are, linguistic values defined by fuzzy sets on the ranges (universes of discourse) X and Y, respectively. The if-part of the rule "*x* is *A*" is called the *antecedent* or premise, while the then-part of the rule "*y* is *B*" is called the *consequent* or conclusion.

In general, the input to an if-then rule is the current value for the input variable and the output is an entire fuzzy set. This set will later be *defuzzified,* assigning one value to the output. [6]

Interpreting an if-then rule involves two steps:

- Evaluation of the antecedent — *Fuzzifying* the inputs and applying any necessary *fuzzy operators*.
- Application of the result to the consequent. is known as *implication.*

2.5.6 Fuzzy Inference

Fuzzy inference can be defined as a process of mapping from a given input to an output, using the theory of fuzzy sets. most inference use as: [5]

2.5.6.i Mamdani Inference

The most commonly used fuzzy inference technique is the so-called Mamdani method. The Mamdani fuzzy inference process is performed in four steps:

fuzzification of the input variables, rule evaluation, aggregation of the rule outputs, and finally defuzzification. [5]

Step1:Fuzzification

The first step is to take the crisp inputs, x1 and y1, and determine the degree to which these inputs belong to each of the appropriate fuzzy sets.

The crisp input is always a numerical value limited to the universe of discourse. In this case, values of x1 and y1 are limited to the universe of discourses X and Y, respectively. The ranges of the universe of discourses can be determined by expert judgements.

Various fuzzy systems use a variety of different crisp inputs. While some of the inputs can be measured directly (height, weight, speed, distance, temperature, pressure etc.), some of them can be based only on expert estimate.

Step 2: Rule Evaluation

The second step is to take the fuzzified inputs, $\mu_{(x=A1)} = 0.5$,$\mu_{(x=A2} = 0.2$, $\mu_{(y=B1)} = 0.1$ and $\mu_{(y=B2)} = 0.7$, and apply them to the antecedents of the fuzzy rules. If a given fuzzy rule has multiple antecedents, the fuzzy operator (AND or OR) is used to obtain a single number that represents the result of the antecedent evaluation. This number (the truth value) is then applied to the consequent membership function.

Step 3: Aggregation of the Rule Outputs

Aggregation is the process of unification of the outputs of all rules. In other words, the membership functions of all rule consequents previously clipped or scaled and combine them into a single fuzzy set. Thus, the input of the aggregation process is the list of

clipped or scaled consequent membership functions, and the output is one fuzzy set for each output variable.

Step 4: Defuzzification

The last step in the fuzzy inference process is defuzzification. Fuzziness helps us to evaluate the rules, but the final output of a fuzzy system has to be a crisp number. The input for the defuzzification process is the aggregate output fuzzy set and the output is a single number.

There are several defuzzification methods, but probably the most popular one is the *centroid technique*. It finds the point where vertical line would slice the aggregate set into two equal masses. [5]

Mathematically this *center of gravity* (COG) can be expressed as:

$$\mathrm{COG} = \frac{\int_a^b \mu_A(x)x\,dx}{\int_a^b \mu_A(x)\,dx} \tag{2.14}$$

The COG is calculated over a continuum of points in the aggregate output membership function, but in practice, a reasonable estimate can be obtained by calculating it over a sample of points. In this case, the following formula is applied:

$$\mathrm{COG} = \frac{\sum_{x=a}^{b} \mu_A(x)x}{\sum_{x=a}^{b} \mu_A(x)} \tag{2.15}$$

2.5.6.ii Sugeno Inference

Mamdani inference, requires to find the centroid of a two-dimensional shape by integrating across a continuously varying function. In general, this process is not computationally efficient.

Can use a singleton, as the membership function of the rule consequent. A singleton, or more precisely a fuzzy singleton, is a fuzzy set with a membership function that is unity at a single particular point on the universe of discourse and zero everywhere else.

Sugeno fuzzy inference is very similar to the Mamdani method. Sugeno changed only a rule consequent. Instead of a fuzzy set, used a mathematical function of the input variable. The format of the Sugeno fuzzy rule is:

IF x is A

AND y is B

THEN z is f (x, y)

where x, y and z are linguistic variables; A and B are fuzzy sets on universe of discourses X and Y, respectively; and f (x, y) is a mathematical function. [5]

The most commonly used zero-order Sugeno fuzzy model applies fuzzy rules in the following form:

IF x is A

AND y is B

THEN z is k

where k is a constant.

The output of each fuzzy rule is constant. In other words, all consequent membership functions are represented by singleton spikes. The similarity of Sugeno and Mamdani methods is quite noticeable. The only distinction is that rule consequents are singletons in Sugeno method.

2.6 DC Motor

The electric motor is a motor that convert electrical energy into mechanical energy. There are two types of motor which are AC motor, and DC motor.

A simple DC motor use electricity and magnetic field for producing torque which rotate the motor. Permanent magnet DC motor (PMDC) outperforms to AC motor because it provides better speed control on high torque loads and use in wide industrial application.

DC motors are more usable as it designed to use with batteries and solar cells energy sources, which provide portability where required it and thus provide cost effective solution, because it is not possible to have AC power supply in every place, DC motor show its response at both voltage and current. The applied voltage describes the speed of motor while current in the armature windings shows the torque.

If applied load increased in the shaft of motor, then in order to sustain its speed motor draws more current from supply and if supply is not able to provide enough current then motor speed will be affected. Generally, it can be said that applied voltage affect speed while torque is controlled by current. DC motors provide more effective results if chopping circuit is used. Low power DC motor usually use in lifting and transportation purposes as low power AC motors do not have good torque capability.

DC motor used in railway engines, electric cars, elevators, robotic applications, car windows and wide verify of small appliances and complex industrial mixing process where torque cannot be compromised.

There are several types of DC motor but most common are brushed DC motor, brushless DC motor, stepper motor, and servo motor. These DC motors have three winding techniques such as shunt DC motor, series DC motor, and compound DC motor.

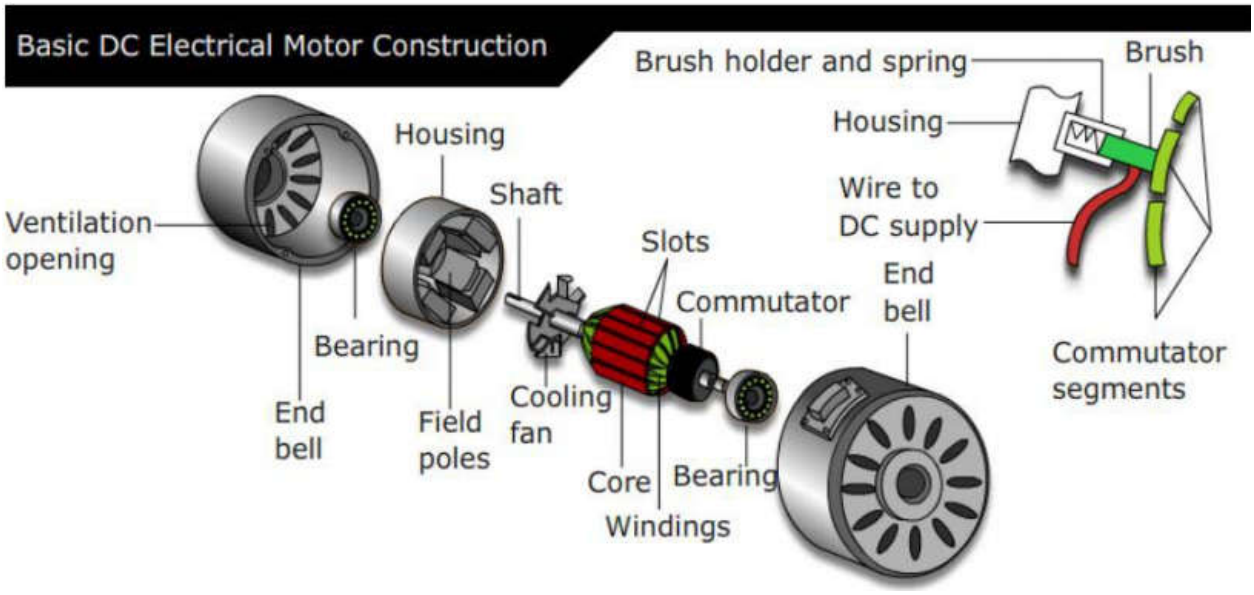


Figure (2.9) DC Motor Parts

The goal in the development of the mathematical model is to relate the voltage applied to the armature to the velocity of the motor. Two balance equations can be developed by considering the electrical and mechanical characteristics of the system. [7]

2.6.1 Electrical Characteristics

The equivalent electrical circuit of a dc motor is illustrated in Figure (2.10). It can be represented by a voltage source (V_a) across the coil of the armature. The electrical equivalent of the armature coil can be described by an inductance (L_a) in series with a resistance (R_a) in series with an induced voltage (V_c) which opposes the voltage source.

The induced voltage is generated by the rotation of the electrical coil through the fixed flux lines of the permanent magnets. This voltage is often referred to as the back emf (electromotive force).

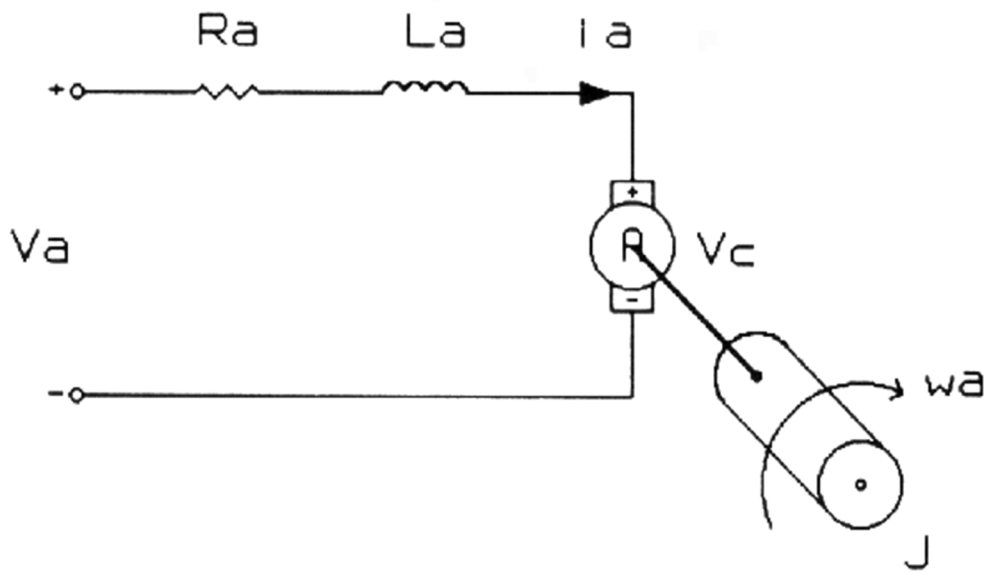


Figure (2.10) Electrical Representation of DC motor.

Differential equation for the equivalent circuit can be derived by using Kirchhoff's voltage law around the electrical loop. Kirchhoff's voltage law states that the sum of all voltages around a loop must equal zero, or

$$V_a - V_{Ra} - V_{La} - V_c = 0 \qquad (2.16)$$

According to Ohm's law, the voltage across the resistor can be represented as

$$V_{Ra} = i_a R_a \qquad (2.17)$$

where i_a is the armature current.

The voltage across the inductor is proportional to the change of current through the coil with respect to time and can be written as

$$V_{La} = L_a \frac{d}{dt} i_a \qquad (2.18)$$

where L_a is the inductance of the armature coil. Finally, the back emf can be written as

$$V_c = k_v \omega_a \qquad (2.19)$$

Where k_v is the velocity constant determined by the flux density of the permanent magnets, the reluctance of the iron core of the armature, and the number of turns of the armature winding. ω_a is the rotational velocity of the armature.

Substituting eqns. (2.17), (2.18), and (2.19) into eqn. (2.16) gives the following differential equation:

$$V_a - i_a R_a - L_a \frac{d}{dt} i_a - k_v \omega_a = 0 \qquad (2.20)$$

2.6.2 Mechanical Characteristics

Performing an energy balance on the system, the sum of the torques of the motor must equal zero. Therefore,

$$T_e - T_{\omega'} - T_\omega - T_L = 0 \qquad (2.21)$$

Where T_e is the electromagnetic torque, $T\omega'$ is the torque due to rotational acceleration of the rotor, $T\omega$ is the torque produced from the velocity of the rotor,

and T_L is the torque of the mechanical load. The electromagnetic torque is proportional to the current through the armature winding and can be written as

$$T_e = k_t i_a \qquad (2.22)$$

Where k_t is the torque constant and like the velocity constant is dependent on the flux density of the fixed magnets, the reluctance of the iron core, and the number of turns in the armature winding. $T_{\omega'}$ can be written as:

$$T_{\omega'} = J\frac{d}{dt}\omega_a \qquad (2.23)$$

Where J is the inertia of the rotor and the equivalent mechanical load. The torque associated with the velocity is written as

$$T_\omega = B\omega_a \qquad (2.24)$$

Where B is, the damping coefficient associated with the mechanical rotational system of the machine.

Substituting eqns. (2.22), (2.23), and (2.24) into eqn. (2.21) gives the following differential equation:

$$k_t i_a - J\frac{d}{dt}\omega_a - B\omega_a - T_L = 0 \qquad (2.25)$$

2.6.3 State Space Representation

The differential equations given in eqns. (2.20) and (2.25) for the armature current and the angular velocity can be written as

$$\frac{d}{dt}i_a = -\frac{R_a}{L_a}i_a - \frac{k_v}{L_a}\omega_a + \frac{V_a}{L_a} \qquad (2.26)$$

$$\frac{d}{dt}\omega_a = \frac{k_t}{J} i_a - \frac{B}{J}\omega_a - \frac{T_L}{J} \qquad (2.27)$$

which describe the dc motor system, Putting the differential equations into state space form gives

$$\frac{d}{dt}\begin{bmatrix} i_a \\ \omega_a \end{bmatrix} = \begin{bmatrix} -\frac{R_a}{L_a} & -\frac{k_v}{L_a} \\ \frac{k_t}{J} & -\frac{B}{J} \end{bmatrix}\begin{bmatrix} i_a \\ \omega_a \end{bmatrix} + \begin{bmatrix} \frac{1}{L_a} & 0 \\ 0 & -\frac{1}{J} \end{bmatrix}\begin{bmatrix} V_a \\ T_L \end{bmatrix} \qquad (2.28)$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} i_a \\ \omega_a \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} V_a \\ T_L \end{bmatrix} \qquad (2.29)$$

which is expressed symbolically as

$$\frac{d}{dt}\underline{x} = \underline{\underline{A}}\,\underline{x} + \underline{\underline{B}}\,\underline{u} \qquad (2.30)$$

$$\underline{y} = \underline{\underline{C}}\,\underline{x} + \underline{\underline{D}}\,\underline{u} \qquad (2.31)$$

where x _is the state vector, u_ is the input vector, and y_is the output vector.

2.6.4 Transfer Function Block Diagram

A block diagram for the system can be developed from the differential equations given in eqns. (2.26) and (2.27). Taking the Laplace transform of each equation gives

$$sI_a(s) - i_a(0) = -\frac{R_a}{L_a} I_a(s) - \frac{k_v}{L_a}\Omega_a(s) + \frac{1}{L_a} V_a(s) \qquad (2.32)$$

$$s\Omega_a(s) - \omega_a(0) = \frac{k_t}{J} I_a(s) - \frac{B}{J}\Omega_a(s) - \frac{1}{J} T_L(s) \qquad (2.33)$$

Around some steady state value are considered, the initial conditions go to zero and all the variables become some change around a reference state, and the equations can be expressed as follows:

$$I_a(s) = \frac{-k_v \Omega_a(s) + V_a(s)}{L_a s + R_a} \qquad (2.34)$$

$$\Omega_a(s) = \frac{-k_t I_a(s) - T_L(s)}{Js + B} \qquad (2.35)$$

The equations can then easily be put into block diagram form. The block diagram obtained from these equations for a permanent magnet dc motor is shown in Figure (2.11).

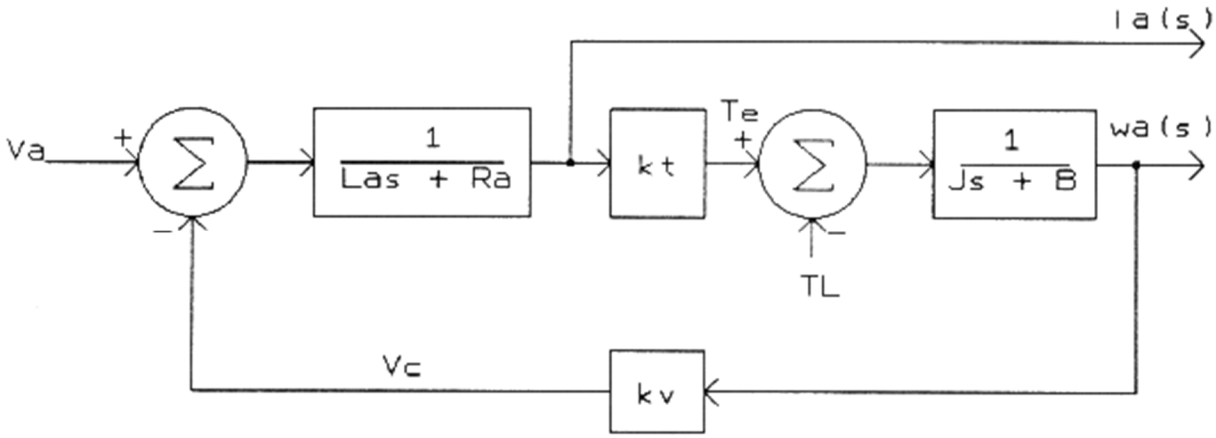


Figure (2.11) Block Diagram Representation of Eqns. (2.34) and (2.35).

The block diagram in Figure (2.11) can be simplified by making the assumption that the load torque is constant. In the case of a sun tracking servo system, the only load torque to be concerned with is the friction in the system, which is relatively constant while the motor is moving. Since the change in T_L is zero, it does not need to appear in the block diagram. Also, if one only focuses on the angular velocity as the response of interest, the block diagram becomes as shown in Figure (2.12).

This block diagram is then easily reduced by block diagram algebra to an overall transfer function. Several steps in this process are shown in Figure (2.13), with the overall transfer function between the output angular velocity and input applied voltage given within the last block in Figure (2.13).

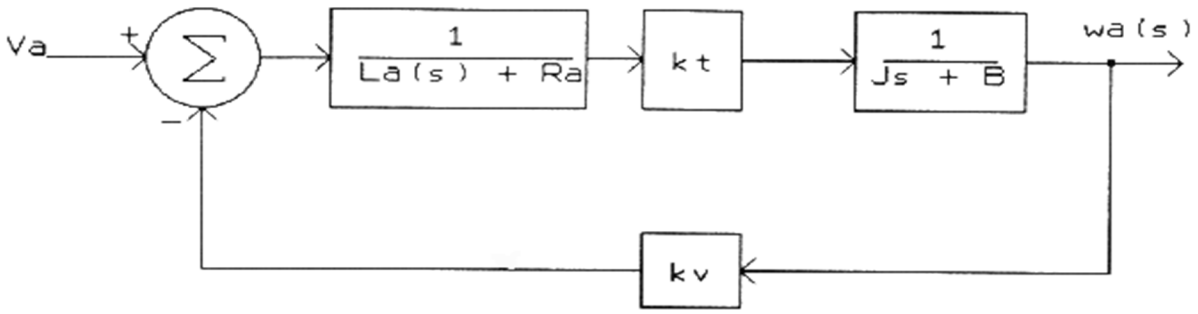


Figure (2.12) Block Diagram of the DC Motor Model.

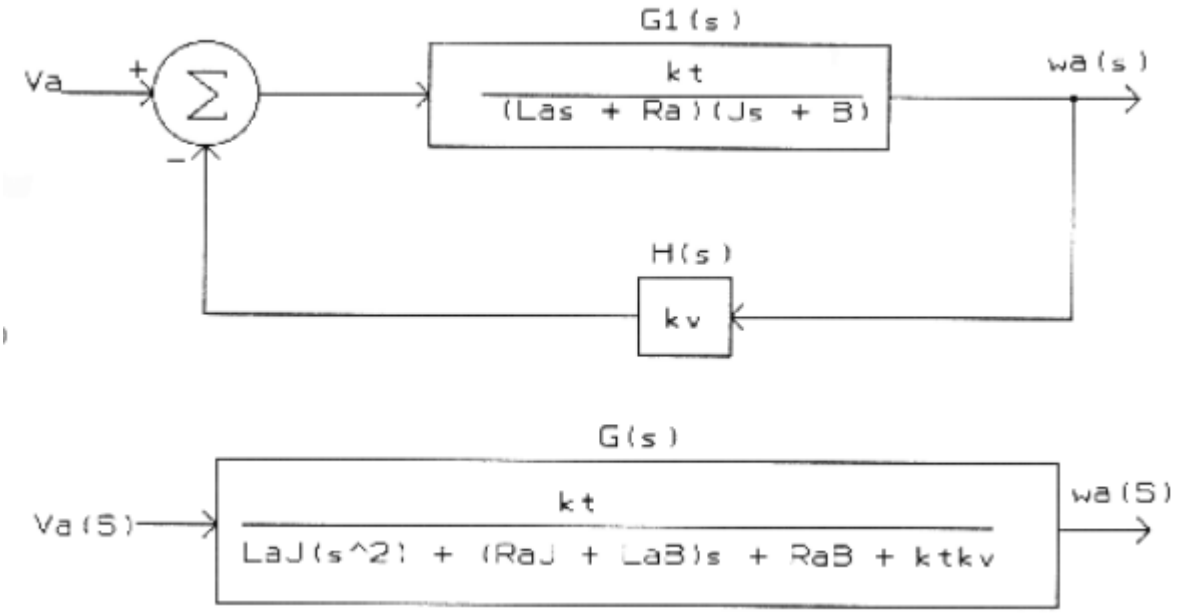


Figure (2.13) Overall Transfer Function for the DC Motor.

2.7 Literature Review

Implementation of self-tuned PID controller (FPID) for speed control of DC motor based on LabVIEW (Laboratory Virtual Instrument Engineering Workbench Environment). speed of a DC motor is controlled using different tuning algorithms. The DC motors are popular in the industry control area for a long time because they possess good characteristics.

The basic property of DC motor is that speed of DC motor can be adjusted by varying its terminal voltage. The real-time application can be easily understood and interfaced with LabVIEW with a quick response. It can easily be deduced that FPID (Fuzzy plus PID) can maintain the speed at desired values irrespective of change of load.

The simulation results demonstrate that the designed self-tuned PID controller realize a perfect speed tracking with lesser rise time and settling time, minimum steady state error and give better performance compared to conventional PID controller. [8]

High performance motor drives are very important in industrial, applying a conventional control algorithm (PI, PD, PID) in a speed controller are the effects of non-linearity in a DC motor. The fuzzy self-tuning approach implemented on a conventional PID structure was able to improve the dynamic as well as the static response of the system.

The simulation results demonstrate that the designed self-tuned PID controller realize a good dynamic behavior of the DC motor, a perfect speed tracking with less rise and settling time, minimum overshoot, minimum steady state error and give better performance compared to conventional PID controller. [9]

DC motors have been widely used in many industrial applications. Speed control is a common requirement in industrial drives in the presence of varying operating conditions i.e. load disturbance, parameter uncertainty, measurement noise etc.

Conventional controllers with fixed parameters have not been successful for real time applications because of the drift in the plant operating conditions. Adaptive techniques are best suited for these situations. The controllers have been considered without modeling of final control element (FCE). In practical applications, the effect of dynamics and nonlinearity of FCE affects the performance of the system, so it is necessary to consider these for a dependable simulation study of the drive performance.

The overall system becomes non-linear due to the dynamics of converter used as FCE. the transfer function of the buck converter is obtained by considering a small linear region near the operating point, then using overall transfer function of buck converter and dc motor, the PID settings are obtained. Then fuzzy logic is used to update these settings on-line corresponding to the changes that may occur in system operating conditions. [10]

Chapter Three

Simulink Model of System

3.1 Simulink Model of DC Motor

The block diagram of a DC motor armature voltage control system is shown in figure (2.13), Simulink model of DC motor shown in figure (3.1).

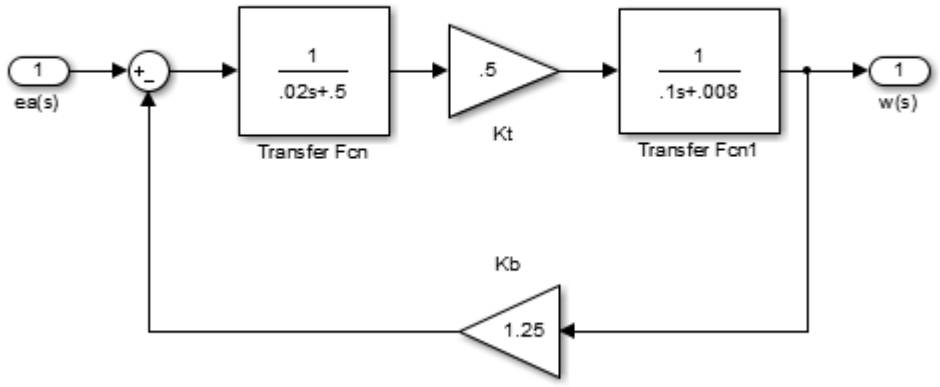


Figure (3.1) Simulink Model of DC Motor

3.2 PID Controller Design

PID controller is mainly to adjust an appropriate proportional gain (K_P), integral gain *(K_I)*, and differential gain (K_D) to achieve the optimal control performance. Ziegler- Nichols is a type of continuous cycling method for controller tuning, *kp* is increased from small value till the point at which the system goes too unstable.

The gain at which system starts oscillating is noted as the ultimate gain *(Ku)* and period of oscillations is the ultimate time period *(Pu),* these two

parameters are used to find the loop-tuning constants of the PID controller using Table (2.2). PID controller system is shown in figure (3.2).

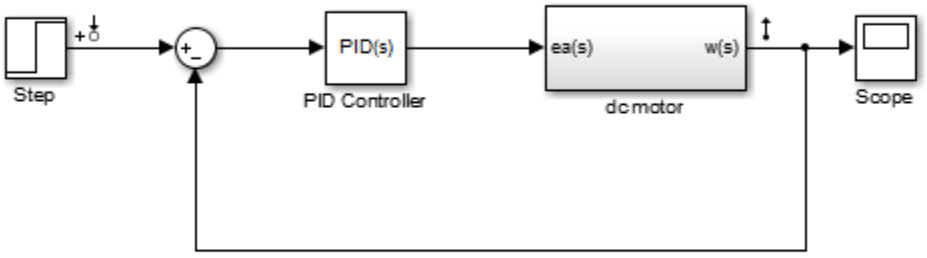


Figure (3.2) Mat lab/Simulink Model of System Using PID Controller.

3.3 Fuzzy Logic Controller Design

Fuzzy logic is express by means of the human language. Based on fuzzy logic, a fuzzy controller converts a linguistic control strategy into automatic control strategy, and fuzzy rules are constructed by expert experience or knowledge database. Fuzzy logic controller (FLC) process is shown in figure (3.3)

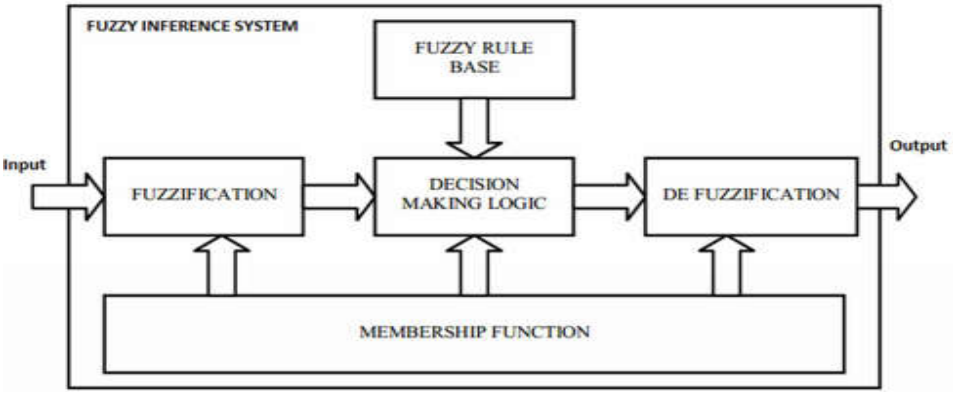


Figure (3.3) Process of Fuzzy Logic Controller

First, set the error *e(t)* and the change of error *ce(t)* of the angular velocity to be the variable inputs of the fuzzy logic controller. The control voltage u(t) is the variable output of the fuzzy logic controller.

$$e(t) = r(t) - u(t) \tag{3.1}$$

$$ce(t) = e(t) - e(t\text{-}1) \tag{3.2}$$

block diagram for speed control of DC motor using fuzzy logic controller is shown in figure (3.4)

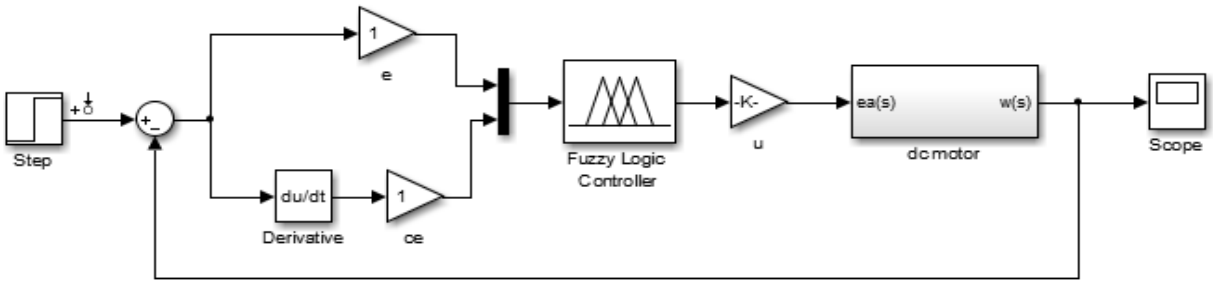


Figure (3.4) Simulink Model of the System Using Fuzzy Logic Controller

The linguistic variables in the Fuzzy inference are defined as {el, em, eh, ecl, ecm, ech, ol, om, oh}, where el means error low, em means error medium, eh means error high, ecl means error change low, ecm means error change medium, ech means error change high, ol means output low, om means output medium and oh means output high.

That shown in membership functions in figure (3.5) for error input and figure (3.6) change of error inputs, in figure (3.7) for output.

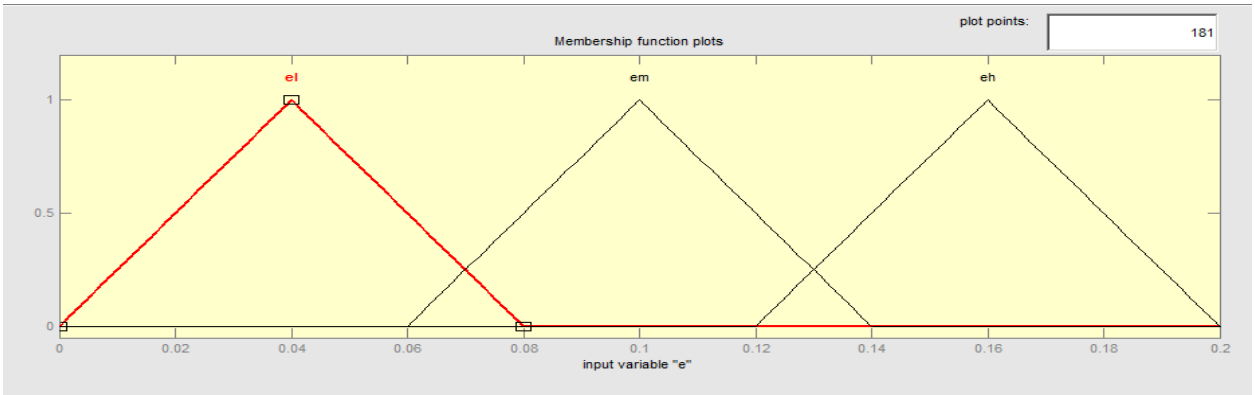


Figure (3.5) Membership Functions for Error Input.

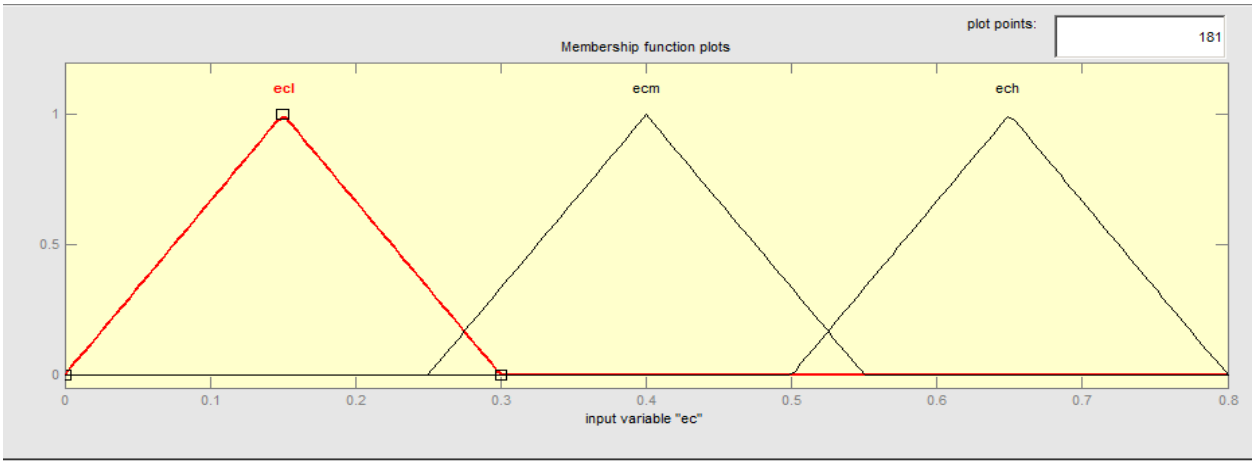


Figure (3.6) Membership Functions for Change of Error Input.

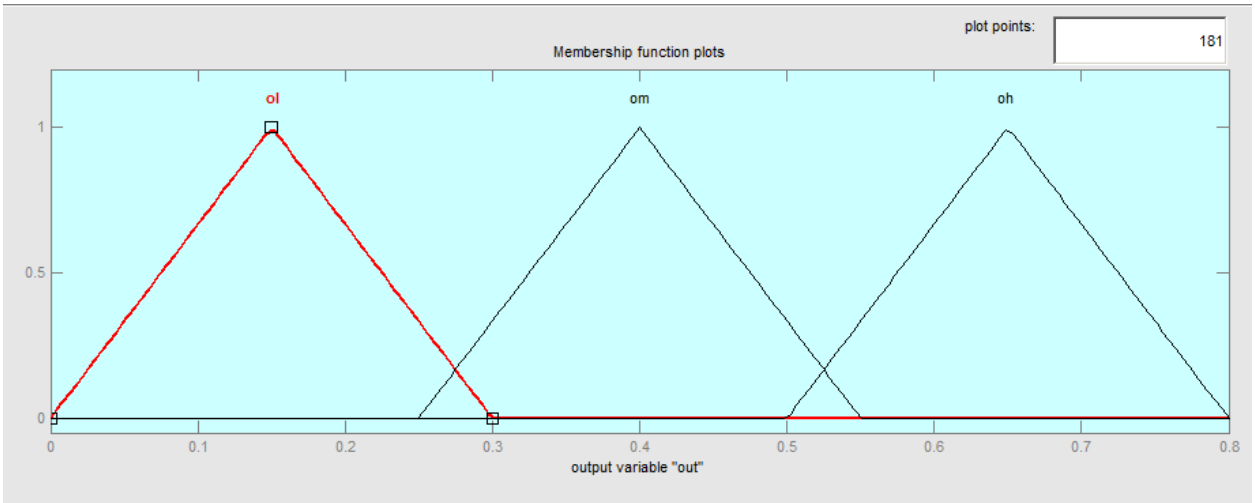


Figure (3.7) Membership Function for Output.

The fuzzy rules are summarized in Table (3.1). the type of fuzzy inference engine is Mamadani.

Table (3.1) Fuzzy Rules

e\ce	*ecl*	*ecm*	*ech*
el	*ol*	*om*	*om*
em	*ol*	*om*	*oh*
eh	*om*	*om*	*oh*

In figure (3.8) fuzzy if-then rules are shown total 9 rules output variable and in figure (3.9) show relation between inputs and output for fuzzy controller.

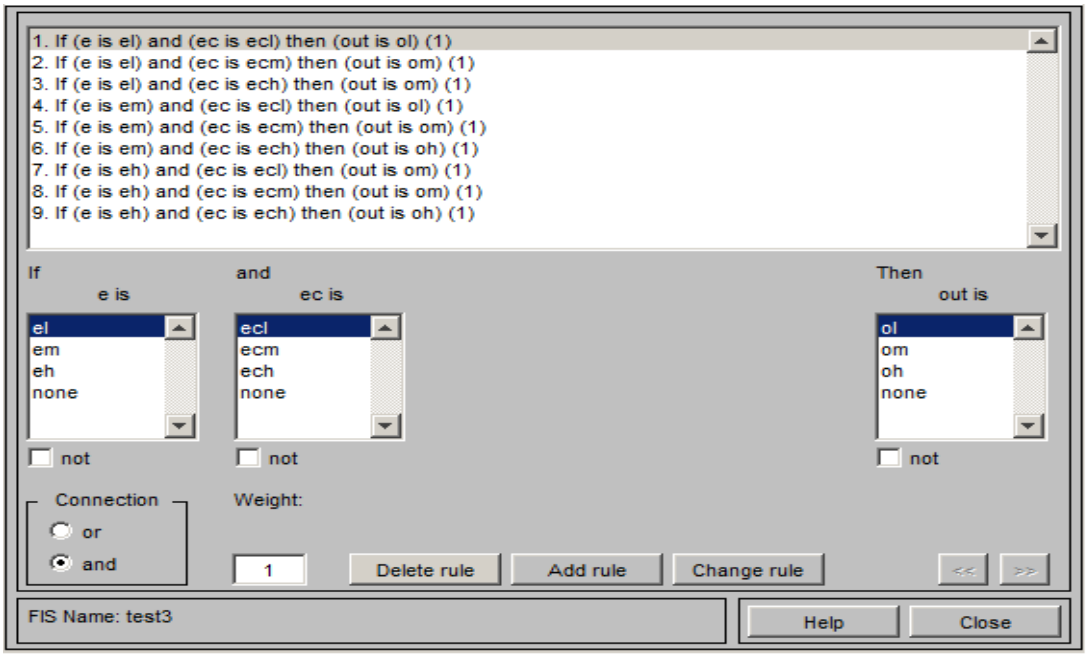


Figure (3.8) Fuzzy IF-Then Rules.

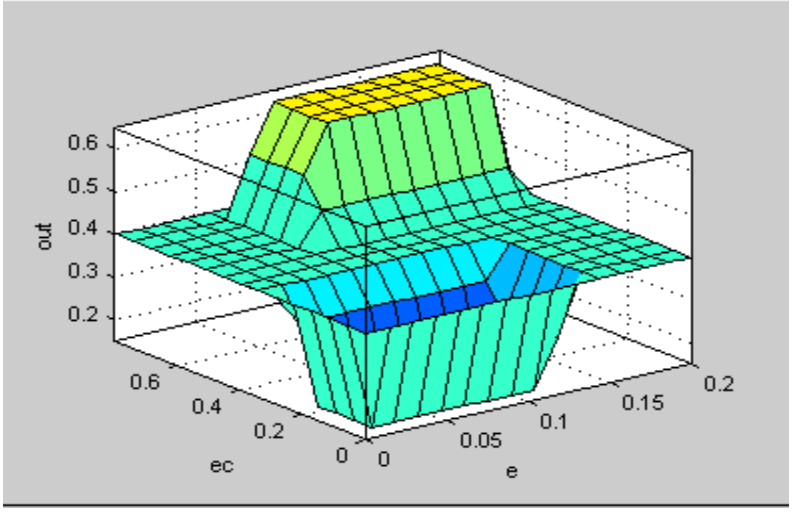


Figure (3.9) Input and Output Relation for Fuzzy Controller.

3.4 Fuzzy-PID Controller Design

The structure of the fuzzy auto tuning PID controller designed for control speed of dc motor is shown in figure (3.10). its inputs are control error *(e)* and the change of control error *(ce)*. the fuzzy auto tuner block adjusts the parameter of the incremental PID controller, and the incremental PID controller calculates the control output.

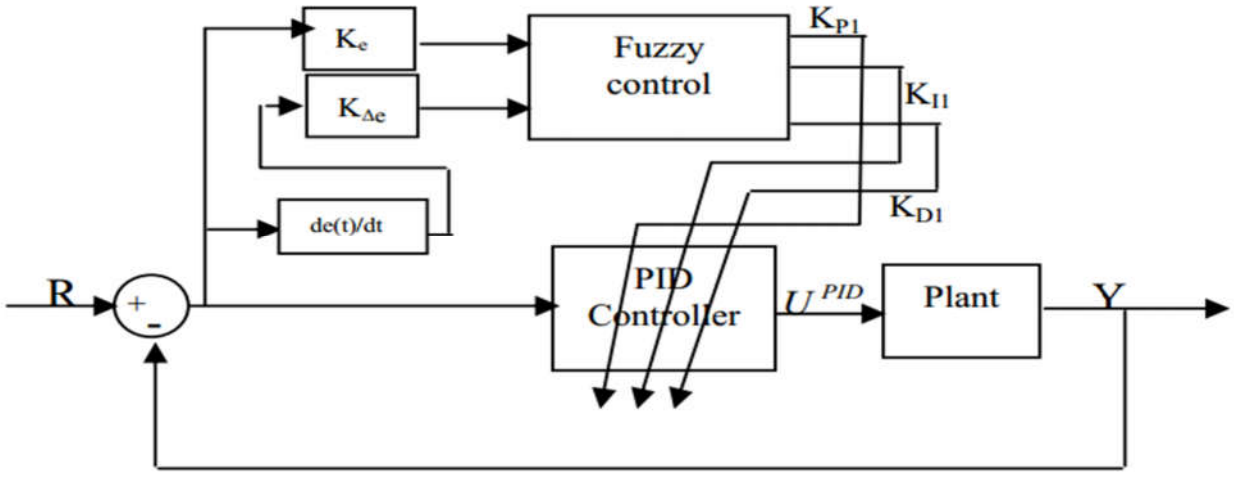


Figure (3.10) The Structure of Self-Tuning Fuzzy-PID Controller

The fuzzy auto-tuning of PID controller is to find the fuzzy logic relationship between three parameters of PID with error *(e)* and change of error *(ce),* calculate *(e)*and *(ce)*in cycle in the operation of control system and adjust *(kp),(ki)* and *(kd)* on-line according to the fuzzy logic control principle .

In this way ,the fuzzy tuner can satisfy the different requirements of PID controller parameters with different *(e)* and *(ce),* and make the controlled object possess good dynamic and static performance .when tuning ,the fuzzy tuner should adjust the value of *(kp), (ki)* and *(kd)* in the PID controller comprehensively taking into consideration of all control specifications including system stability ,response speed, overshoot, steady-state control error etc.

The role of proportional control action is to speed up the control response of controlled system ,and to improve control accuracy .the integral control action is taken to eliminate the steady state control error ,and the role of differential control action is to improve the dynamic property of control system .

The universe of discourse of fuzzy variable sets including speed control error *(e),* change of error *(ce)* are defined as :NL means negative large, NS means negative small, ZE means zero, PS means positive small, PL means positive large, and *kp* (the change of *kp*), *ki* (the change of *ki*) and *kd* (the change of *kd*) are defined as :PVS means positive very small, PS means positive small, PMS means positive medium small, PM means positive medium, PML means positive medium large, PL means positive large, PVL means positive very large.

The fuzzy variable sets including speed control error *(e),* error change *(ce), (kp), (ki)* and *(kd)* are defined as follows:

$$e = \{-20, 40, 70, 100, 160\} \quad (3.3)$$

$$ce = \{-1300, -800, -550, -300, 200\} \quad (3.4)$$

$$kp = \{0, 5, 10, 15, 20, 25, 30\} \quad (3.5)$$

$$ki = \{0, 10, 20, 30, 40, 50, 60\} \quad (3.6)$$

$$kd = \{0, 1, 2, 3, 4, 5,6\} \quad (3.7)$$

Set the membership function of fuzzy variables *(e), (ce), (kp), (ki)* and *(kd)* as triangle membership function (trimf), the degree of membership function of speed control error *(e)* is shown in figure (3.11), the degree of membership of change speed error *(ce)* is shown in figure (3.12), the degree of membership of change *kp (kp)* is shown in Figure (3.13), the degree of membership of change *ki (ki)* is shown in Figure (3.14), the degree of membership of change *kd (kd)* is shown in Figure (3.15).

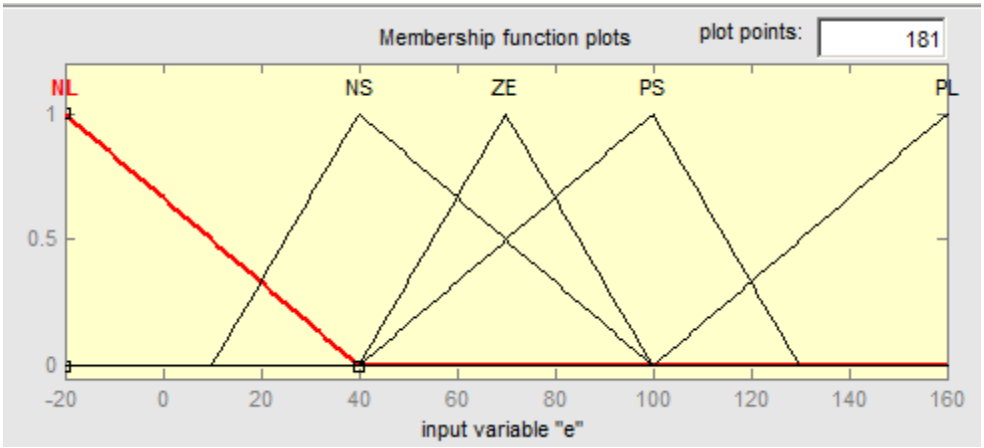


Figure (3.11) Degree of Membership of Speed Control Error (e).

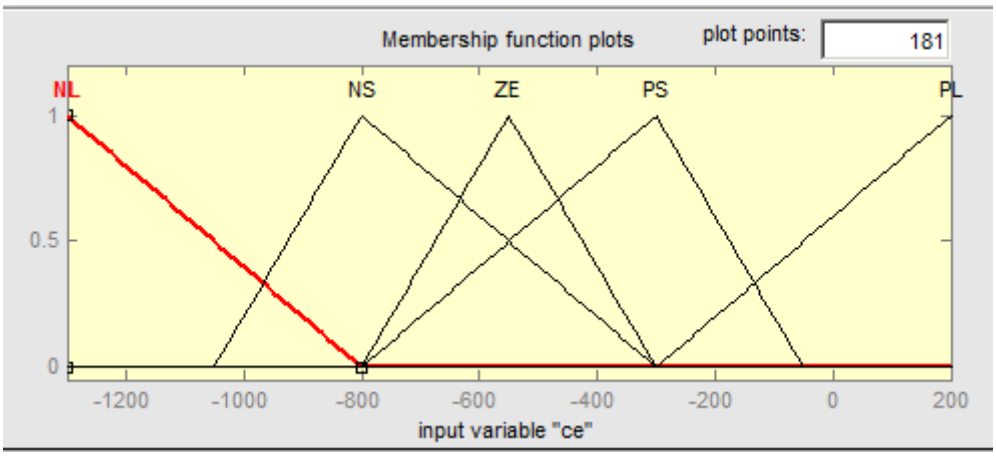


Figure (3.12)Degree of Membership of Change Speed Error (ce).

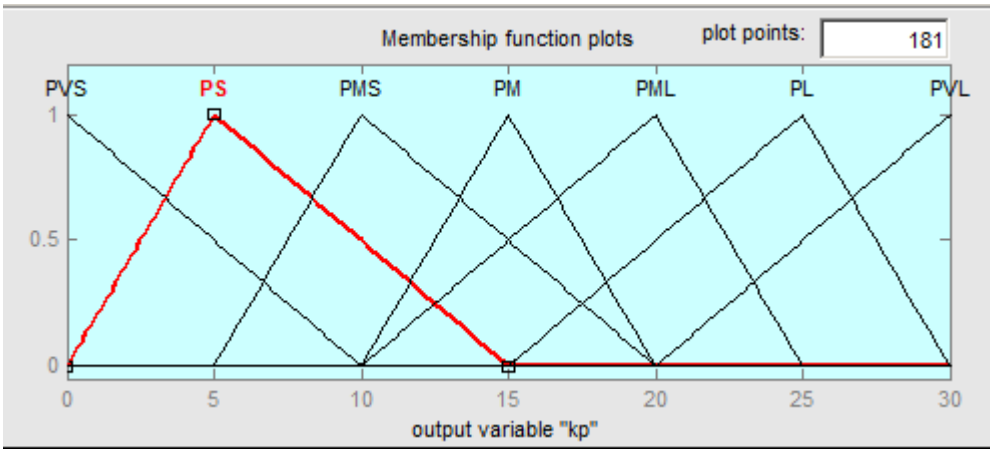


Figure (3.13)Degree of Membership of Change k_p (k_p).

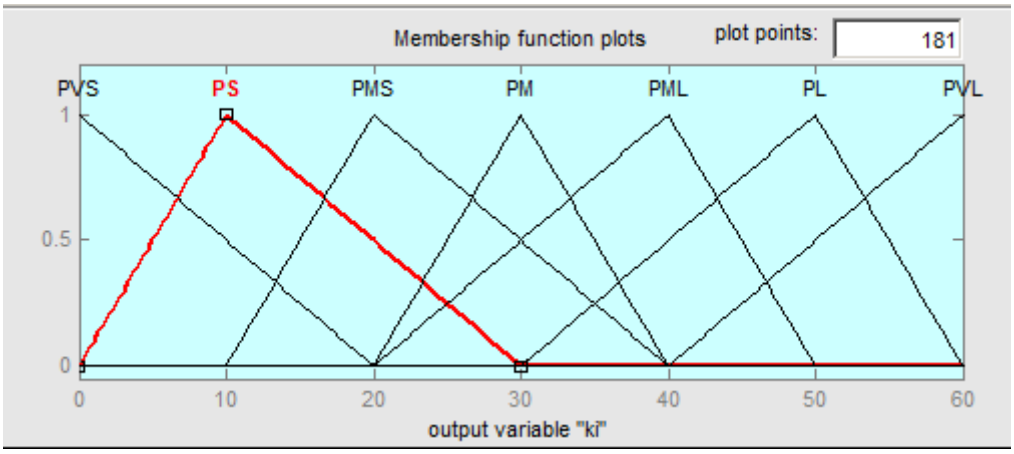


Figure (3.14) Degree of Membership of Change k_i (k_i).

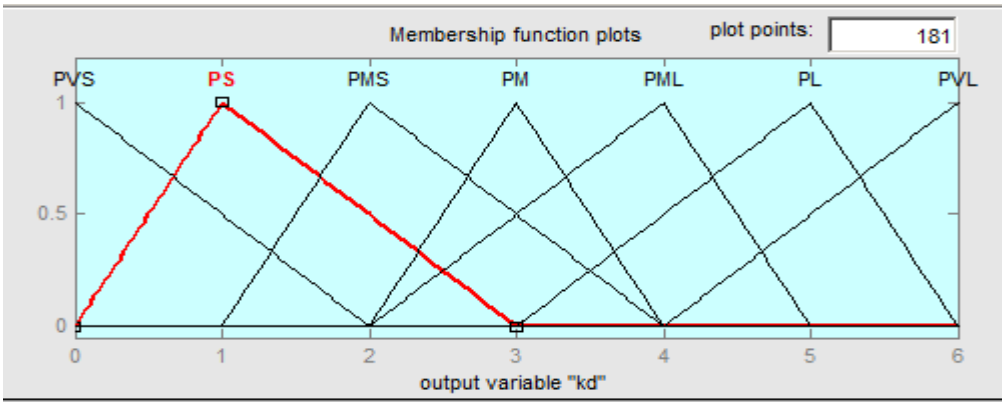


Figure (3.15) Degree of Membership of Change k_d (k_d).

The fuzzy tuning rule sets of *(kp), (ki)* and *(kd)* are designed as illustrated in table (3.2) (3.3) and (3.4) as follows:

Table (3.2) Fuzzy Tuning Rule of (k_p).

ce/e	*NL*	*NS*	*ZE*	*PS*	*PL*
NL	*PVL*	*PVL*	*PVL*	*PVL*	*PVL*
NS	*PML*	*PML*	*PML*	*PL*	*PVL*
ZE	*PVS*	*PVS*	*PS*	*PMS*	*PMS*
PS	*PML*	*PML*	*PML*	*PL*	*PVL*
PL	*PVL*	*PVL*	*PVL*	*PVL*	*PVL*

Table (3.3) Fuzzy Tuning Rule of (k_i).

ce/e	*NL*	*NS*	*ZE*	*PS*	*PL*
NL	*PM*	*PM*	*PM*	*PM*	*PM*
NS	*PMS*	*PMS*	*PMS*	*PMS*	*PMS*
ZE	*PS*	*PS*	*PVS*	*PS*	*PS*
PS	*PMS*	*PMS*	*PMS*	*PMS*	*PMS*
PL	*PM*	*PM*	*PM*	*PM*	*PM*

Table (3.4) Fuzzy Tuning Rule of (k_d).

ce/e	*NL*	*NS*	*ZE*	*PS*	*PL*
NL	*PVS*	*PMS*	*PM*	*PL*	*PVL*
NS	*PMS*	*PML*	*PL*	*PVL*	*PVL*
ZE	*PM*	*PL*	*PL*	*PVL*	*PVL*
PS	*PML*	*PVL*	*PVL*	*PVL*	*PVL*
PL	*PVL*	*PVL*	*PVL*	*PVL*	*PVL*

In figure (3.16) fuzzy if-then rules are shown total 25 rules outputs variable and in figure (3.17) show relation between inputs and outputs for fuzzy controller.

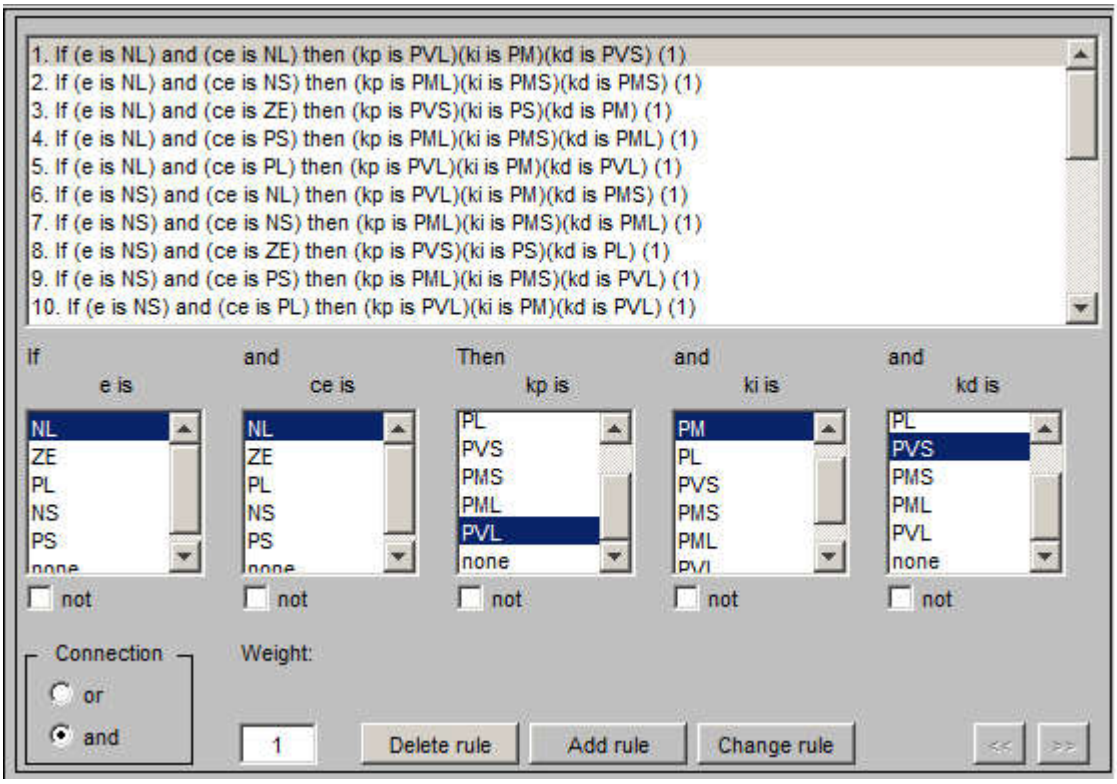
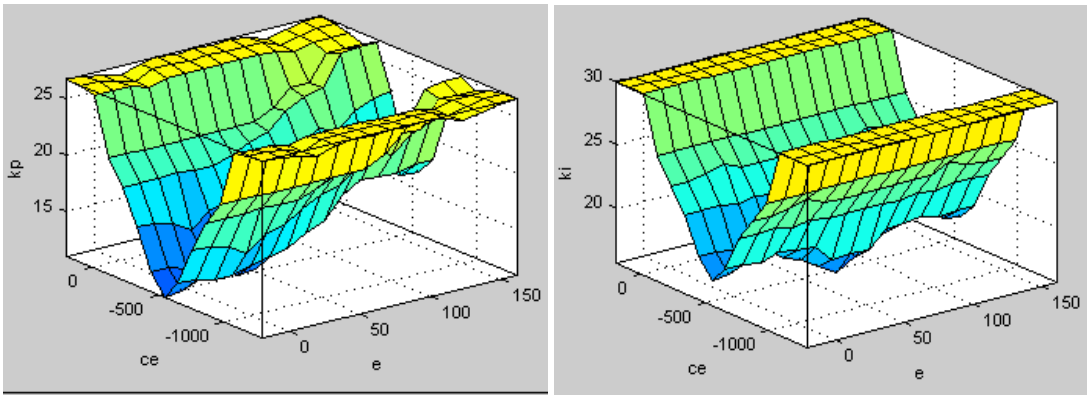
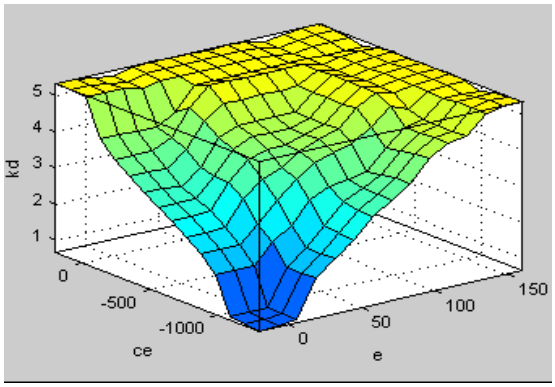


Figure (3.16) Fuzzy IF-Then Rules.



(a) ***(b)***



(c)

Figure (3.17) Relation Between Inputs and Outputs (a) Surface Viewer of (k_p) (b)Surface Viewer of (k_i) (c)Surface Viewer of (k_d).

According to the membership degree calculation table of all fuzzy variable sets, and the fuzzy tuning rule sets and algorithms of all control parameters, the mamdani-type inference system Fuzzy-PID is developed using the fuzzy control simulation toolbox" fuzzy "of mat lab. The simulation model for Fuzzy-PID self-tuning controller is shown in figure (3.18) it contains 2 input *(e, ce)*,3 outputs *(k_p, k_i and k_d)* and 25 fuzzy inference rules.

Figure (3.18)Simulink Model for Fuzzy-PID Self-Tuning Controller

Self-tuning fuzzy-PID regulator subsystem block is shown in figure (3.19) (3.20) Consists of fuzzy and PID block with some modification refers to the formula which is applied to calibrate the value of *(kp), (ki)* and *(kd)* from fuzzy block to obtain the value of *(kp), (ki)* and *(kd).* Each parameter has it is own calibration.

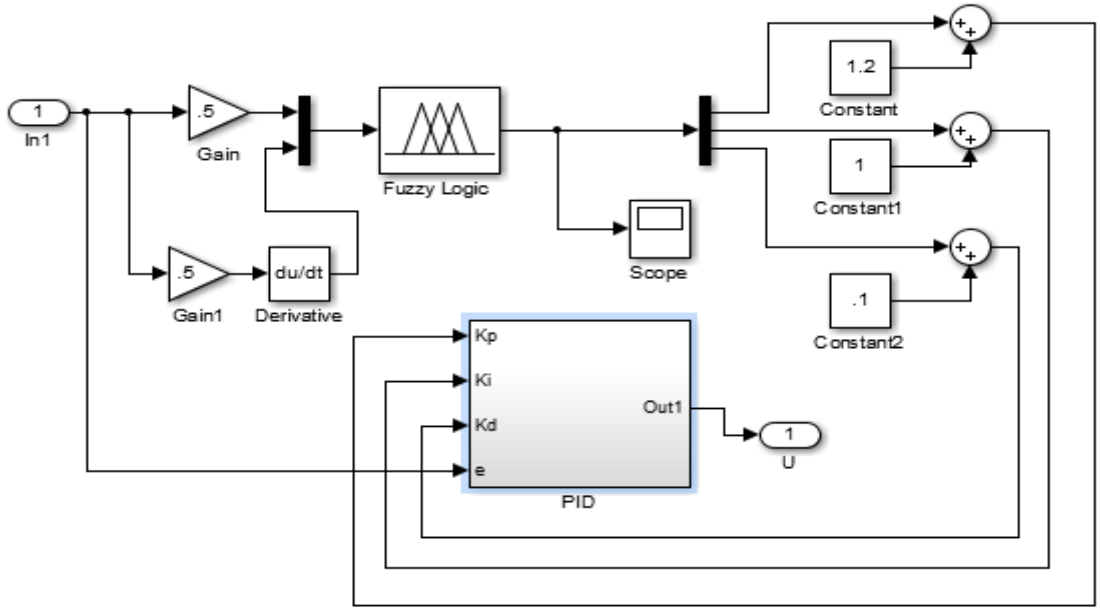


Figure (3.19) Simulation Block of Fuzzy PID Self-Tuning.

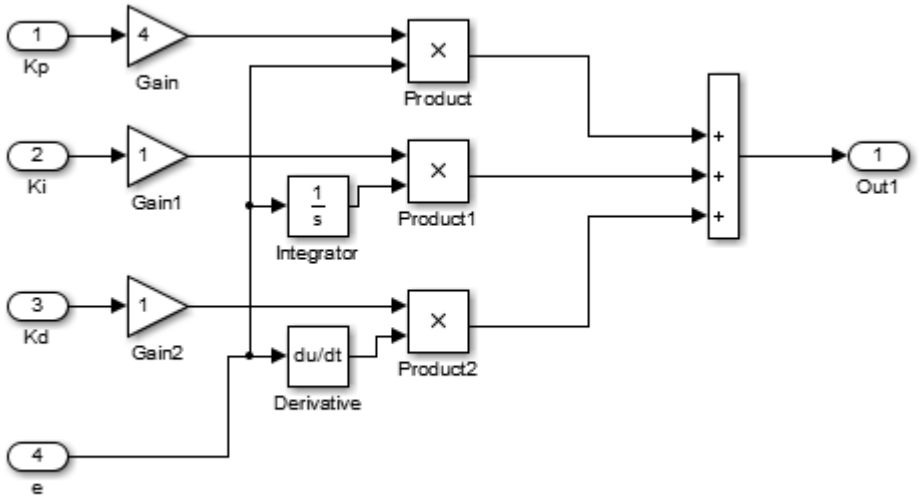


Figure (3.20) Simulation Block of Fuzzy PID Self-Tuning.

After run the Simulink block automatically to carry out the fuzzy auto tuning of PID parameters, through looking up the fuzzy tuning rule tables. Then fuzzy PID will calculate the tuned new values of *kp, ki* and *kd*, using the following algorithm:

$$K_p = k_p 0 + k_p \quad (3.8)$$

$$K_i = k_i 0 + k_i \quad (3.9)$$

$$K_d = k_d 0 + k_d \quad (3.10)$$

Where *kp0, ki0* and *kd0* are constants.

Chapter Four

Result and Discussion

After simulating the process control of speed control system models by using MATLAB/SIMULINK, the performance of the controllers was analyzed by comparing the output signal represented by the graph.

Figures (4.1) show performance of the PID controller. Using Ziegler_Nichols method to tune PID controller parameters. Figure (4.2) show Error of the closed loop system using PID controller.

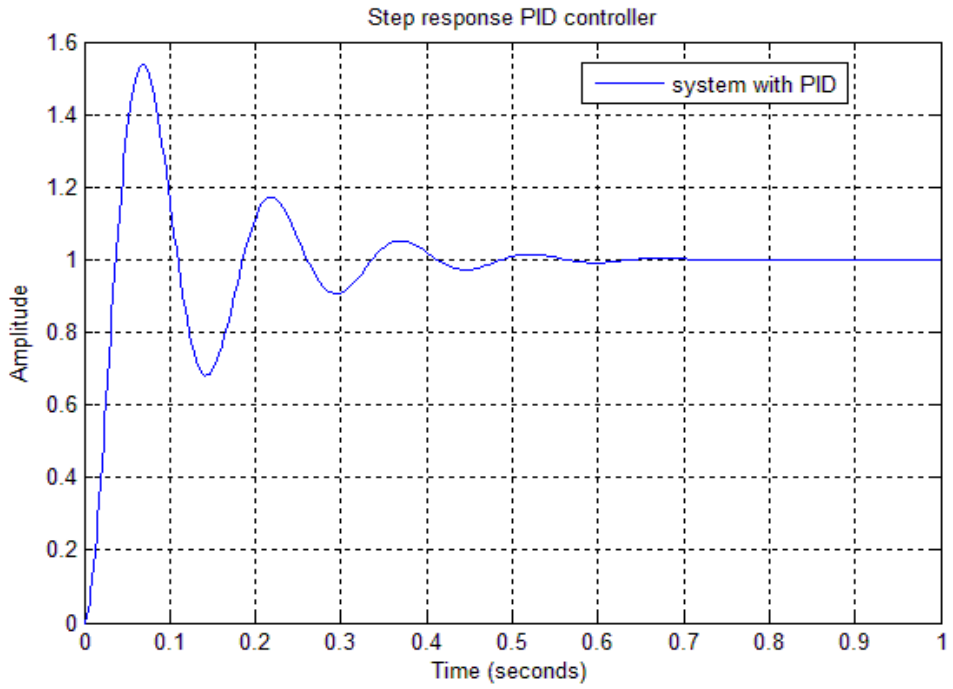


Figure (4.1) Step Response of the System with PID Controller.

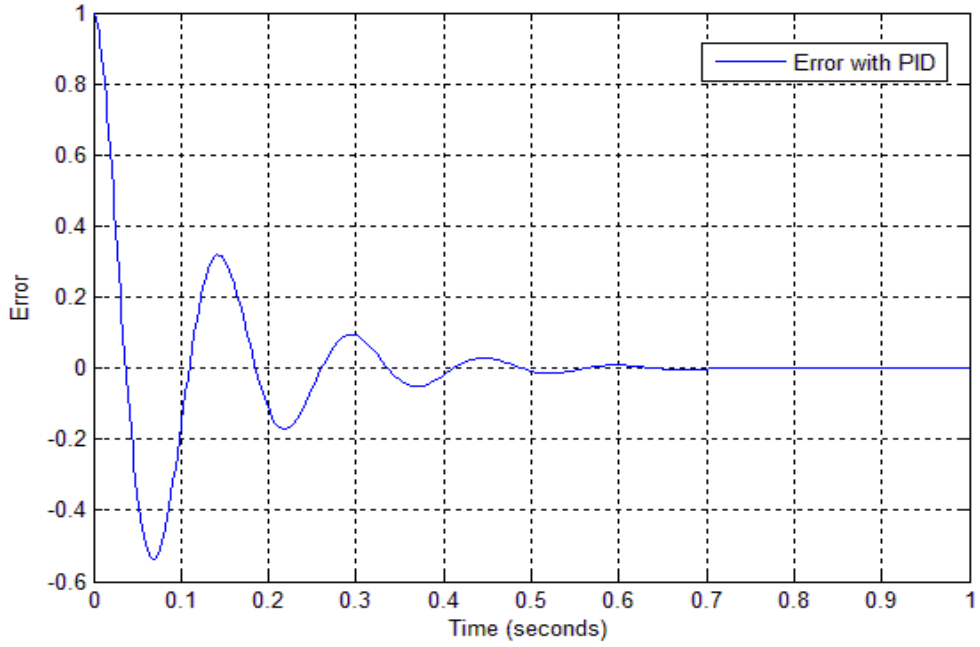


Figure (4.2) Error of the Closed Loop System Using PID Controller

Figures (4.3) show performance of the Fuzzy controller. Figure (4.4) show Error of the closed loop system using Fuzzy controller.

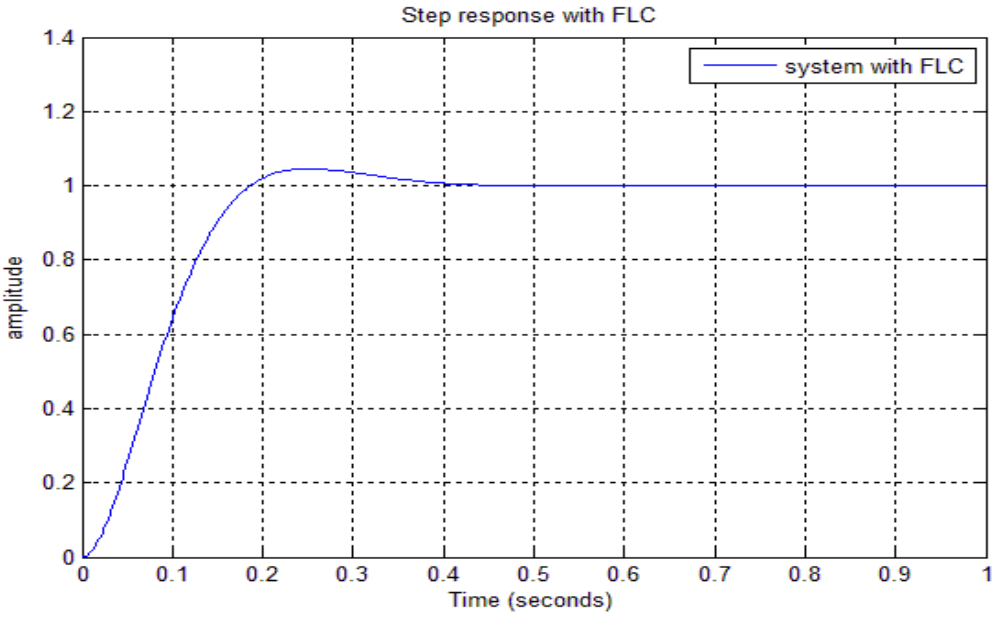


Figure (4.3) Step Response of the System with Fuzzy Controller.

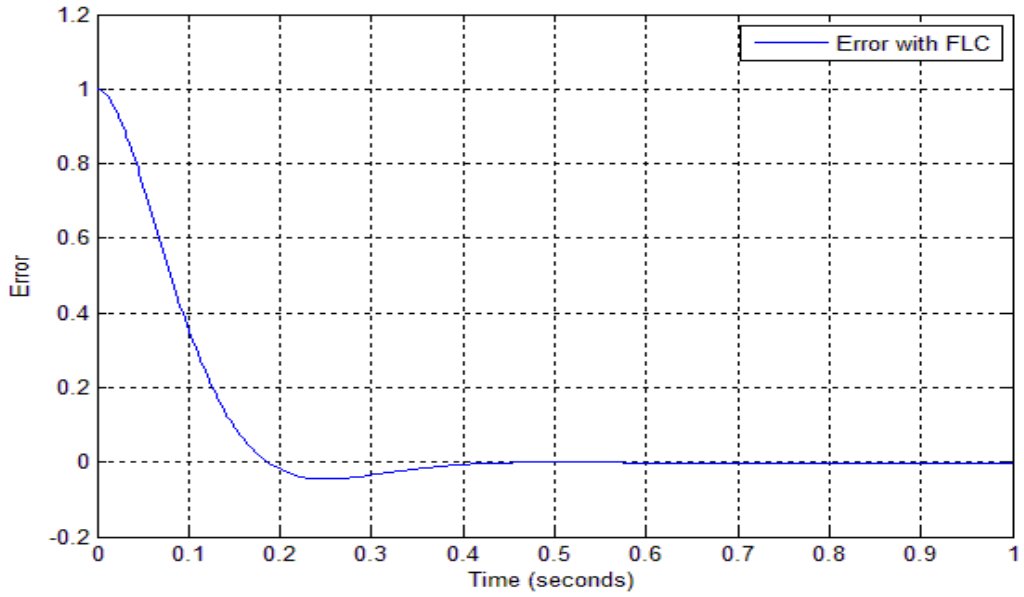


Figure (4.4) Error of the Closed Loop System Using Fuzzy Controller

Figures (4.5) show step response of system using fuzzy-PID controller. Figure (4.6) show Error of the closed loop system using fuzzy-PID controller.

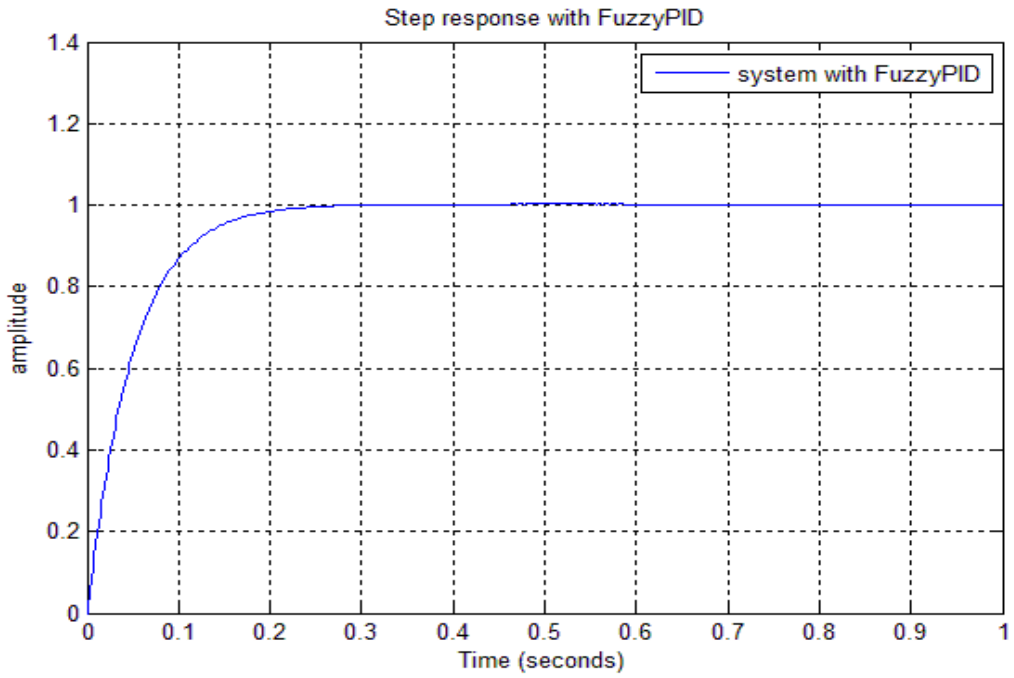


Figure (4.5) Step Response of System Using Fuzzy-PID Controller

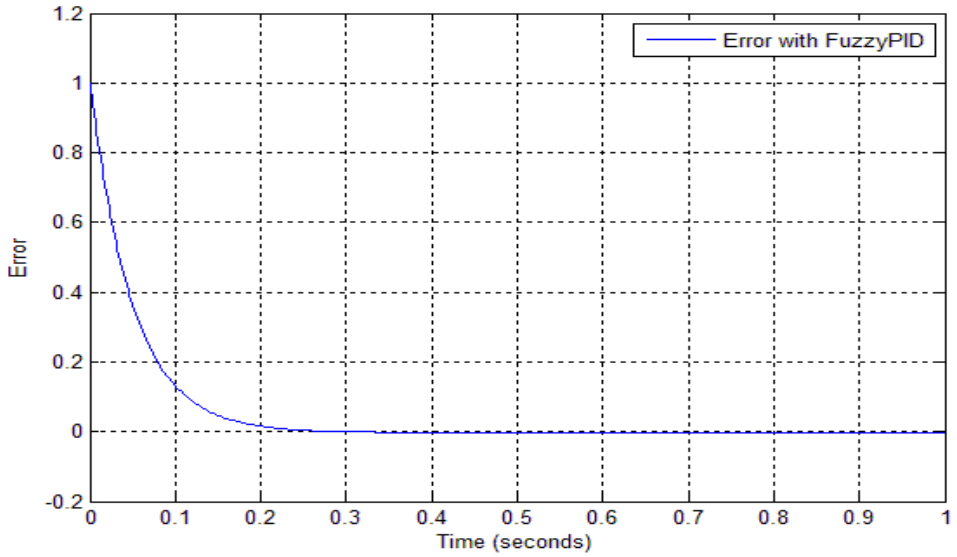


Figure (4.6) Error of the Closed Loop System Using Fuzzy-PID Controller

Comparison between PID, Fuzzy and Fuzzy plus PID controller step response specification as shown in figure (4.7) and table (4.1).

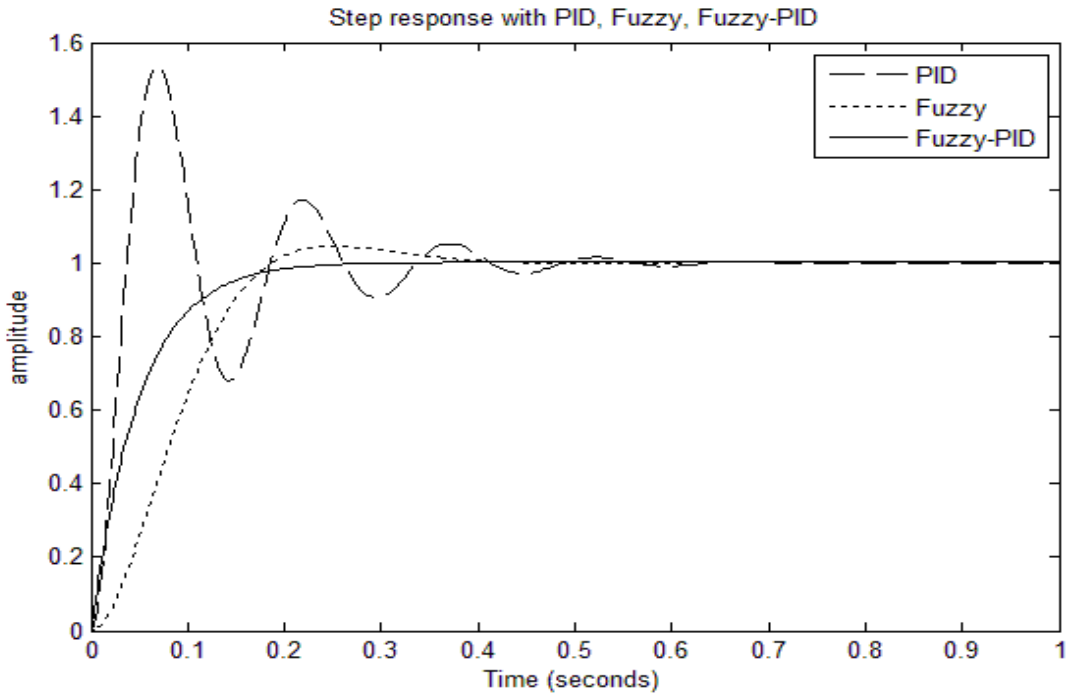


Figure (4.7) Step Response of System.

Table (4.1) Comparison Between the Output Responses for Controller

Title	PID controller	Fuzzy controller	Fuzzy-PID controller
Peak amplitude	*1.48*	*1.045*	*1*
Peak time (sec)	*0.0742*	*0.25*	*0.29*
% Overshoot	*48.3*	*4.5*	*0*
Settling time (sec)	*0.571*	*0.36*	*0.2*
Rise time (sec)	*0.03*	*0.15*	*0.11*

From the simulation results it is concluded that, compared with the conventional PID controller, Fuzzy controller and self-tuning PID controller. Fuzzy-PID controller has a better performance in both transient and steady state response. The self-tuning Fuzzy-PID has better dynamic response curve, short response time, smaller overshoot, less peak amplitude, minimum settling time, small steady state error (SSE), high steady precision compared to the conventional PID controller and Fuzzy controller.

Chapter Five

Conclusion and Recommendation

5.1 Conclusion

In this project designed a DC motor whose speed can be controlled using PID controller. The proportional, integral and derivate *(KP, KI, KD)* gains of the PID controller are adjusted according to FUZZY LOGIC. First, the fuzzy logic controller is designed according to fuzzy rules so that the systems are fundamentally robust.

There are 25 fuzzy rules for self-tuning of each parameter of PID controller. The FLC has two inputs. One is the motor speed error between the reference and actual speed and the second is change in speed error (speed error derivative). Secondly, the output of the FLC i.e. the parameters of PID controller are used to control the speed of DC Motor.

The study shows that both characters of PID controllers and characters of fuzzy controller are present in fuzzy self-tuning PID controller. The fuzzy self-tuning approach implemented on a conventional PID structure was able to improve the dynamic as well as the static response of the system.

Comparison between the conventional output and the fuzzy self-tuning output was done on the basis of the simulation result obtained by MATLAB. The simulation results demonstrate that the designed self-tuned PID controller realize a good dynamic behavior of the DC motor, a perfect speed tracking with less rise and settling time, minimum overshoot, minimum steady state error and give better performance compared to conventional PID controller.

5.2 Recommendations

- Fuzzy PID self-tuning was set for SISO system through this research, similar one can be designed for MIMO system to evaluate its response.
- This technique can be extended to other types of motors.
- The parameters of PID controller can also be tuned by using genetic algorithm (GA).

References

[1] Rao, V.Dukkipati," Analysis and Design of control systems using MATLAB", 2006.

[2] Karl Johan A strom," Control System Design", 2002.

[3] https://en.wikipedia.org/wiki/PID_controller#PID_controller_theory**.**

[4] Franck dernoncourt, "introduction to fuzzy logic", mit, jan 2013.

[5] Michael Negnevitsky, "Artificial Intelligence", second edition:2005.

[6] Fuzzy Logic Toolbox™ User's Guide.

[7] http://www.profjrwhite.com/system_dynamics/sdyn /s6/s6fmathm/s6fmathm.

[8] Salim, J. O," Fuzzy Based PID Controller for Speed Control of D.C. Motor Using LabVIEW" October 2015.

[9] Umesh. K. B and R. N" Speed Control of DC Motor Using Fuzzy PID Controller"2013.

[10] Ravishankar. K, Dhaval. R" Fuzzy Adaptive PID Speed Control of a Converter Driven DC Motor" April-2015.

[11] Satya Sheel, Omhari Gupta" High Performance Fuzzy Adaptive PID Speed Control of a Converter Driven DC Motor" March, 2012.

[12] karl j. astrom, tore .h" PID controller: theory, design and tuning"2nd ed.

[13] Philip A. Adewuyi," DC Motor Speed Control: A Case between PID Controller and Fuzzy Logic Controller" MAY 2013.

[14] Husain. A, Gagan. S, Vikash. B, Saket. S, Shubham. A"Controlling of DC motor using Fuzzy Logic Contoller" (CAC2S 2013).

[15] Rekha.K,Sulochana :"Speed Control of Separately Exited DC motor using Fuzzy Logic Controller"(IJETT) 2013.

Appendix

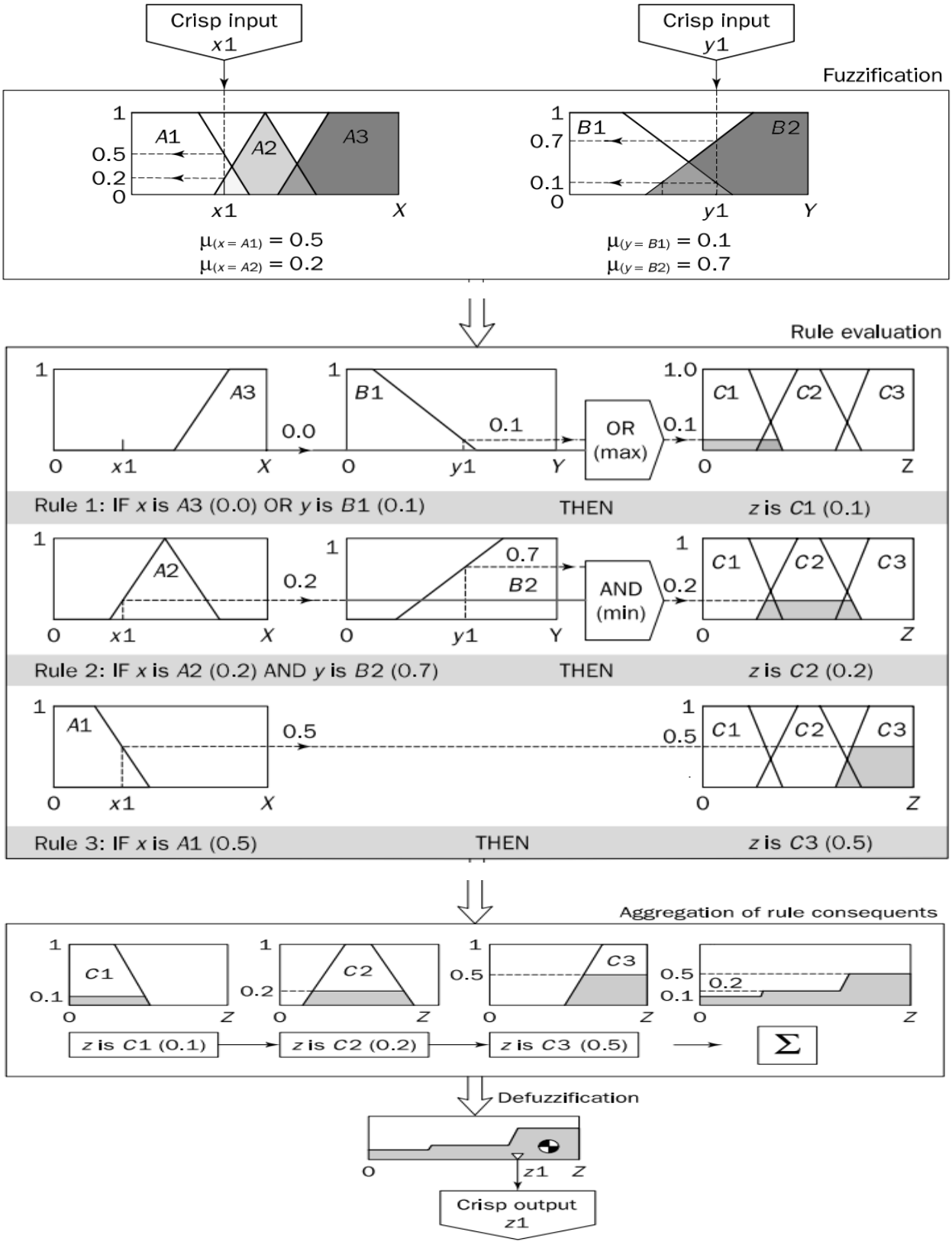
➢ *Mamdani Fuzzy Inference*

To see how everything fits together, a simple two-input one output problem that includes three rules:

Rule: 1	**Rule: 1**
IF x is A3	**IF** ***project funding is adequate***
OR y is B1	**OR** ***project staffing is small***
THEN z is C1	**THEN** ***risk is low***
Rule: 2	**Rule: 2**
IF x is A2	**IF** ***project funding is marginal***
AND y is B2	**AND** ***project staffing is large***
THEN z is C2	**THEN** ***risk is normal***
Rule: 3	**Rule: 3**
IF x is A1	**IF** ***project funding is inadequate***
THEN z is C3	**THEN** ***risk is high***

where x, y and z (project funding, project staffing and risk) are linguistic variables; A1, A2 and A3 (inadequate, marginal and adequate) are linguistic values determined by fuzzy sets on universe of discourse X (project funding); B1 and B2 (small and large) are linguistic values determined by fuzzy sets on universe of discourse Y (project staffing); C1, C2 and C3 (low, normal and high) are linguistic values determined by fuzzy sets on universe of discourse Z (risk).

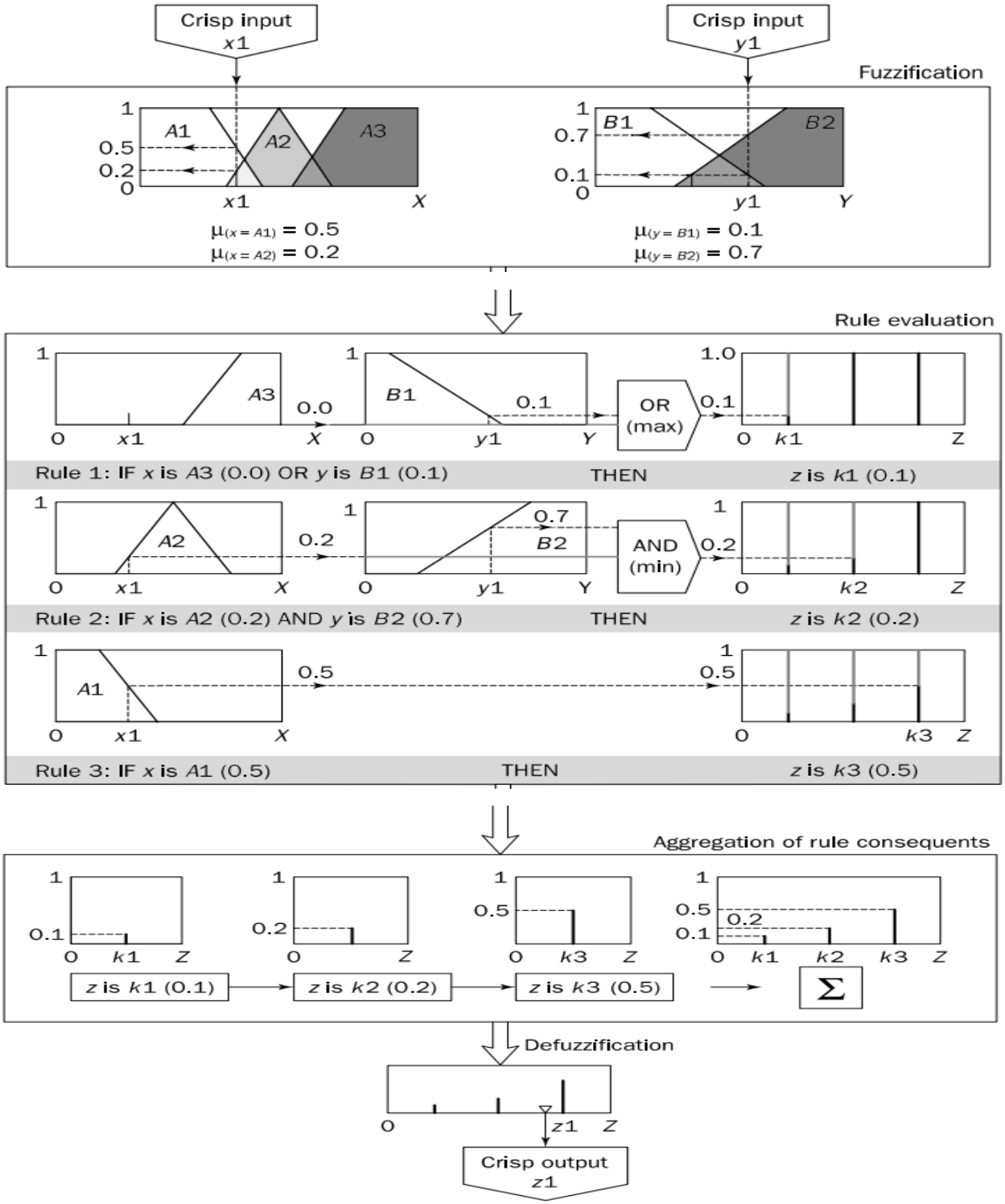
The basic structure of Mamdani fuzzy inference is shown in Figure



The basic structure of Mamdani fuzzy inference

- ***Sugeno Inference***

Figure shows the fuzzy inference process for a zero-order Sugeno model.



The basic structure of Sugeno fuzzy inference

- ***DC Motor Model Parameters***

Armature resistance (R_a) = 0.5 Ω

Torque constant (K_T) = 0.5 Nm/A

Armature inductance (L_a) = 0.02 H

Friction constant (B) = 0.008 Nms/rad

Rotor inertia (J) = 0.1 kgm^2

Back emf constant (K_b) = 1.25 Vs/rad