DELHI TECHNOLOGICAL UNIVERSITY

EE-206
Control Systems

Project Report

Project Title
DC Motor Speed Control Using Adaptive Fuzzy-PID Controller

Submitted To:
Prof. Madhusudan Singh
Professor
Dept. Electrical Engineering

Submitted By:
Abhigyan Adarsh 2K19/EE/005
Anirudh Singh 2K19/EE/042

CONTENTS

ABSTRACT

We have designed and simulated a modified PID control using an adaptive Fuzzy controller applied to the speed control of a DC motor modeled in Simulink. An Armature Controlled DC motor has been modeled in Simulink using its actual parameters to reproduce real world results. The PID controller has been designed using plant modeling and the PID tuner toolbox provided in MATLAB. An adaptive Fuzzy logic controller for this modified PID has been designed using the Fuzzy Logic Toolbox. The rule base for the fuzzy logic has been made using reference from various research works cited at the end. The complete system consisting of adaptive fuzzy PID controller along with DC motor model is simulated using MATLAB/Simulink. A comparison of conventional PID with adaptive fuzzy PID controller is also demonstrated for reference speed tracking and load disturbance. The results produced show that the adaptive fuzzy tuned PID controller provides better dynamic behavior of DC motor with low rise time, low settling time, minimum overshoot and low steady-state error in speed and thus provides superior performance.

1. INTRODUCTION

Engineers rely on motion-control devices to improve efficiencies and production rates on automated factory floors, or at least maintain them. The DC motor is one of the first machines devised to convert electrical power into mechanical power motors. DC motors are widely used in industrial applications, robot manipulators and home appliances, because of their high reliability, flexibility and low cost, where speed and position control of motor are required.

PID controllers are commonly used for motor control applications because of their simple structures and intuitively comprehensible control algorithms. Proportional-integral-derivative (PID) control offers a straightforward and most effective method to many real world problems with its three-term functionality that can handle both the transient and steady-state responses. Thus, PID controllers are very commonly exploited in conventional speed control loops [4]. Conventional PID controllers have some shortcomings like, undesirable overshoot and stagnant response due to sudden change in load torque and sensitivity to controller gains. The volatile behaviors of motor parameters during operating conditions and existence of noise in the system are the two main problems in motor control. So, it becomes a challenging task of motor control of complex, non- linear and time varying systems using conventional methods. Thus, it is an obvious requirement to tune the parameters of the PID controller to attain optimal state under field conditions. An adaptive fuzzy logic based PID controller design is suggested to get better results than conventional one [3, 6].

With conventional speed control techniques, desired speed tracking with acceptable accuracy in case of abrupt disturbances and parameter variation cannot be achieved and this problem can be suppressed by execution of advanced control techniques such as adaptive control, fuzzy logic control (FLC). Fuzzy control theory endows a nonlinear controller which can perform complex nonlinear control action even for indefinite nonlinear systems [7].FLC is a prevailing tool for controlling the parameters of a system. The fuzzy set concept was introduced by Zadeh in1965. The FLC can be used as two forms, first one as a part of a control loop in the system and the second one as a supervisor. The second form, i.e. supervisory can be used for adjusting the coefficients of PID controller and this technique is termed as fuzzy gain scheduling [8].

The FLC based gain scheduling of PID controller works as an adaptive PID controller and can be used with non-linear systems to get superior results. In order to demonstrate the potency of FLC to obtain an accurate tracking, an arm robot manipulator model verifies the controller by simulation in [9].

1.1) DC Motor

A common actuator in control systems is the DC motor. It directly provides rotary motion and coupled with wheels or drums and cables, can provide translational motion. The DC motor uses electricity and a magnetic field to produce torque, which causes it to turn. It requires two magnets of opposite polarity and an electric coil, which acts as an

electromagnet. The repellent and attractive electromagnetic forces of the magnets provide the torque that causes the motor to turn. It also consists of one set of coils, called armature winding, inside a set of permanent magnets, called the stator. Applying a voltage to the coils produces a torque in the armature, resulting in motion. In a DC motor, several such coils are wound on the rotor, all of which experience force, resulting in rotation. The greater the current in the wire, or the greater the magnetic field, the faster the wire movement because of the greater force created. At the same time, torque is being produced as the conductors are moving in a magnetic field.

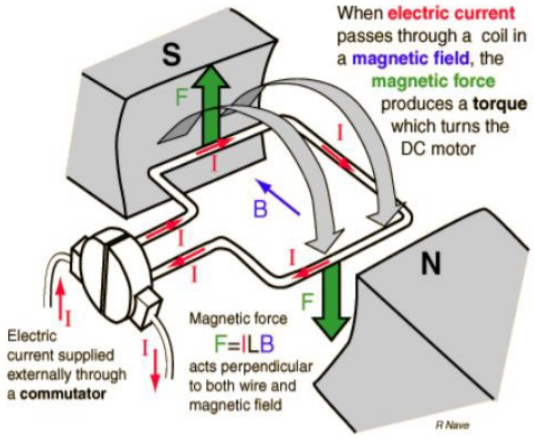


Fig. 1. Torque Production in a DC Motor

At different positions, the flux linked with it changes, which causes an electromagnetic force (EMF) to be induced as shown in Figure 2. This voltage is in opposition to the voltage that causes current flow through the conductor and is referred to as a counter voltage or back EMF. The value of current flowing through the armature is dependent upon the difference between the applied voltage and this counter-voltage.

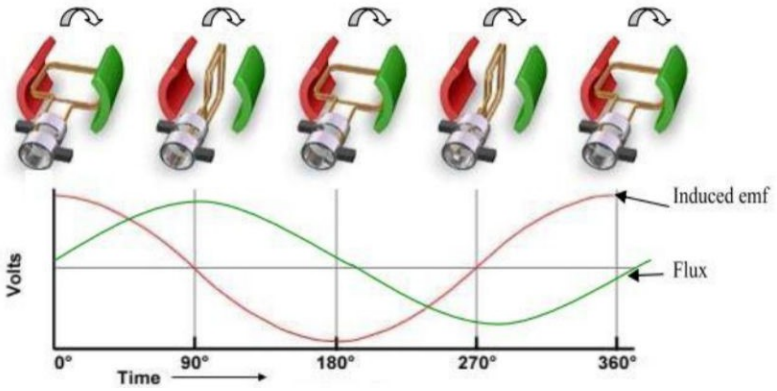


Fig. 2. Induced voltage in Armature of a DC Motor

Current due to this counter-voltage tends to oppose the very cause for its production according to Lenz's law. It results in the rotor slowing down such that the force created by the magnetic field equals the load force applied on the shaft. Then the system moves at constant velocity. Basically, the operation of a DC motor is based on the principle that when a current carrying conductor is placed in a magnetic field, the conductor experiences a force. There are three methods of controlling the speed of a DC motor: armature voltage speed control, field flux speed control and voltage control. In this study, the armature method was employed.

1.2) Armature Controlled DC Motor: Physical System

A DC motor can be modeled as an electrical circuit made up of a resistance, inductor and voltage sources representing the driving voltage and the back EMF. Consider a DC motor, whose electric circuit of the armature and the free body diagram of the rotor are as shown in the figure 3:

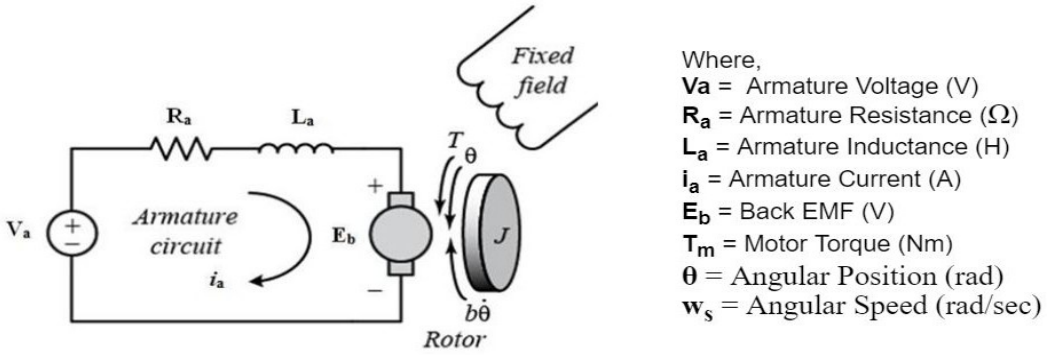


Fig 3. Electrical Circuit Model of an Armature controlled DC Motor

1.3) System Equations

For an armature controlled DC motor, the field excitation is kept unvarying and the armature voltage V_a is varied so that the current through the armature also varies. The motor output torque varies proportionally with the armature current at constant field current, so:

$$T_m = K_t . i_a \qquad (1)$$

From the equivalent circuit of motor, voltage balance equation for armature circuit can be depicted as:

$$V_a = i_a R_a + L_a \frac{di_a}{dt} + E_b \qquad (2)$$

$$E_b = K_b . \omega \qquad (3)$$

Laplace transform of armature voltage equation gives,

$$V_a(s) - E_b(s) = (R_a + L_a.s).I_a(s) \quad (4)$$

$$V_a(s) - K_e.\omega(s) = (R_a + L_a.s)I_a(s) \quad (5)$$

Similar to equation of voltage balance, equation of torque balance for inertial load can be written as follows

$$T_m = J\frac{d\omega}{dt} + B\omega \quad (6)$$

$$T_m(s) = s.J\omega(s) + B\omega(s) \quad (7)$$

$$\frac{\omega(s)}{T_m(s)} = \frac{1}{s.J + B} \quad (8)$$

Thus, from the above equations, we can represent a dc motor using block diagram schema as shown in Figure 4.

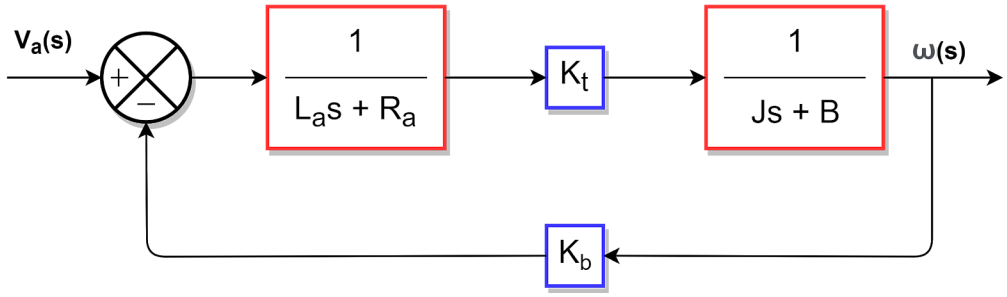


Fig. 4. Block Diagram Representation of a DC Motor

By reducing the block diagram, the transfer function of the motor can be obtained as (9)

$$\frac{\omega(s)}{V_a(s)} = \frac{K_t}{(s.L_a + R_a)(s.J + B) + K_t.K_b} \quad (9)$$

2. PID CONTROLLER

PID is an acronym for the mathematical terms *Proportional, Integral*, and *Derivative*.
A proportional integral derivative (PID) controller can be used as a means of controlling temperature, pressure, flow and other process variables. As its name implies, a PID controller combines proportional control with additional integral and derivative adjustments which help the unit automatically compensate for changes in the system.

The purpose of a PID controller is to force feedback to match a setpoint, such as a thermostat that forces the heating and cooling unit to turn on or off based on a set temperature. PID controllers are best used in systems which have a relatively small mass and those which react quickly to changes in the energy added to the process. It is recommended in systems where the load changes often and the controller is expected to compensate automatically due to frequent changes in setpoint, the amount of energy available, or the mass to be controlled.

The PID Controller can be represented as a block diagram as shown in the figure 5.

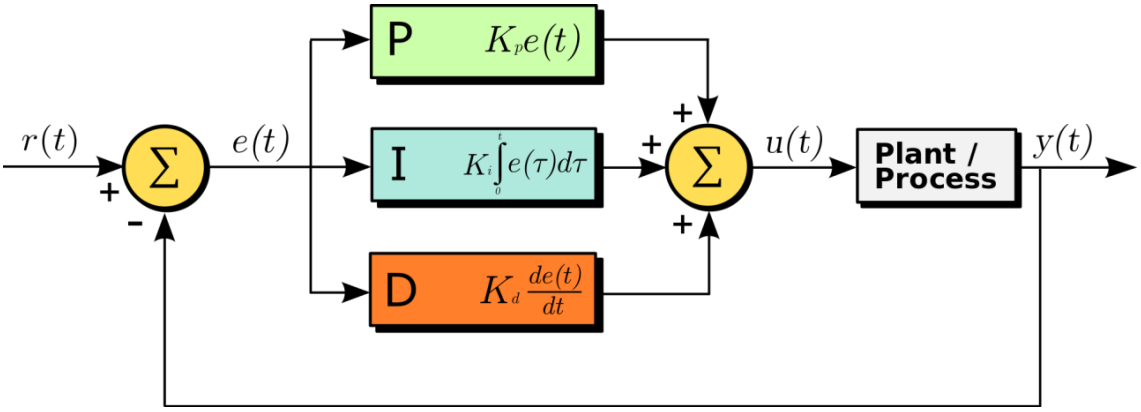


Fig. 5. Block Diagram Representation of a PID Controller

Here, the Plant process represents the system that has to be controlled to achieve a setpoint using the PID Controller. The block diagram shows how the three control terms: Proportional, Integral, and Derivative are generated and applied.
It shows a PID Controller which continuously calculates an error value **e(t)** as the difference between a desired setpoint **SP = r(t)** and a measured process variable **PV = y(t):e(t) = r(t) – y(t)**, and applies a correction based on the **P, I, and D** terms. The controller attempts to minimize the error over time by adjustment of a control variable **u(t).**

The overall system equation for u(t) is given by:

$$u(t) = K_p e(t) + K_i \int e(t)dt + K_p \frac{de}{dt} \qquad (10)$$

2.1) Working of a PID Controller

The working principle behind a PID controller is that the proportional, integral and derivative terms must be individually adjusted or "tuned." Based on the difference between these values a correction factor is calculated and applied to the input. For example, if an oven is cooler than required, the heat will be increased. The three tuning factors and their significance have been explained below:

Proportional Tuning: The output of the proportional factor is the product of gain and measured error ε. Hence, larger proportional gain or error makes for greater output from the proportional factor. Setting the proportional gain too high causes a controller to repeatedly overshoot the setpoint, leading to oscillation. The downside to a proportional-only loop is that when error becomes too small, loop output becomes negligible. Therefore, even when the proportional loop reaches steady state, there is still error. The larger the proportional gain, the smaller the steady state error — but the larger the proportional gain, the more likely the loop is to become unstable. This dilemma leads to inevitable steady-state error called *offset*.

Integral Tuning: Integral tuning attempts to remove this “offset” by effectively cumulating the error result from the "P" action to increase the correction factor. For example, if the oven remained below temperature, “I” would act to increase the head delivered. However, rather than stop heating when the target is reached, "I" attempts to drive the cumulative error to zero, resulting in an overshoot. In fact, most control loop action at steady state is due to the integral factor. Controllers that feature integral reset prove it: Resetting the integral when a loop is in steady state causes controller output to momentarily drop to zero as the integral “basket” is emptied. The downside to the integral factor is that it strongly contributes to controller output overshoot past the target setpoint. The shorter the integral time, the more aggressively the integral works.

Derivative Tuning: A majority of PID loops in the real world are really just PI loops. That does not negate the fact that there are certain applications in which the derivative plays a very important role. The proportional corrects instances of error, the integral corrects accumulation of error, and the derivative corrects present error versus error the last time it was checked. In other words, the derivative is looking at the rate of change of the error Δε. The more error changes or the longer the derivative time, the larger the derivative factor becomes. The effect of the derivative is to counteract the overshoot caused by P and I. When the error is large, the P and I will push the controller output. This controller response makes error change quickly, which in turn causes the derivative to more aggressively counteract the P and the I. A properly used derivative allows for more aggressive proportional and integral factors. Larger derivative time makes the derivative more aggressively dampen P and I.

3. FUZZY LOGIC CONTROL

Fuzzy logic is an approach to computing based on "degrees of truth" rather than the usual "true or false" (1 or 0) Boolean logic on which the modern computer is based. The idea of fuzzy logic was first advanced by Dr. Lotfi Zadeh of the University of California at Berkeley in the 1960s. Dr. Zadeh was working on the problem of computer understanding of natural language. Natural language (like most other activities in life and indeed the universe) is not easily translated into the absolute terms of 0 and 1. (Whether everything is ultimately describable in binary terms is a philosophical question worth pursuing, but in practice much data we might want to feed a computer is in some state in between and so, frequently, are the results of computing.)

It may help to see fuzzy logic as the way reasoning really works and binary or Boolean logic is simply a special case of it. Fuzzy Logic is a particular area of concentration in the study of Artificial Intelligence and is based on the value of that information which is neither definitely true nor false. The information which humans use in their everyday lives to base intuitive decisions and apply general rules of thumb can and should be applied to those control situations which demand them. Acquired knowledge can be a powerful weapon to combat the undesired effects of the system response. For an example consider the following example shown in figure.

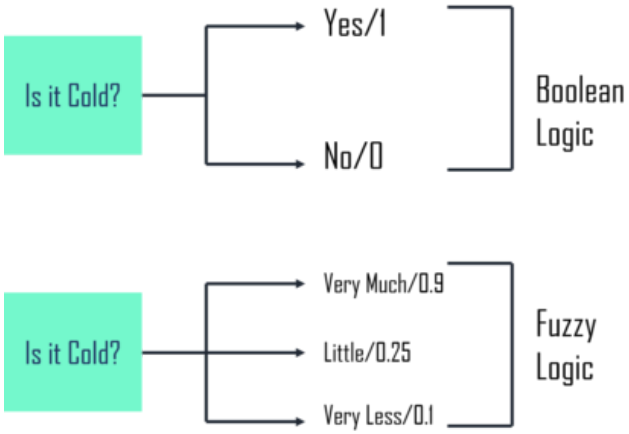


Fig. 6. Example of Fuzzy Logic

Fuzzy logic seems closer to the way our brains work. We aggregate data and form a number of partial truths which we aggregate further into higher truths which in turn, when certain thresholds are exceeded, cause certain further results such as motor reaction. A similar kind of process is used in neural networks, expert systems and other artificial intelligence applications. Fuzzy logic is essential to the development of human-like capabilities for AI, sometimes referred to as artificial general intelligence: the representation of generalized human cognitive abilities in software so that, faced with an unfamiliar task, the AI system could find a solution.

Fuzzy logic Control is applied with great success in various control applications. Almost all the consumer products have fuzzy control. Some of the examples include

controlling your room temperature with the help of air-conditioner, anti-braking system used in vehicles, control on traffic lights, washing machines, large economic systems, etc.
A control system is an arrangement of physical components designed to alter another physical system so that this system exhibits certain desired characteristics. Following are some reasons of using Fuzzy Logic in Control Systems −

- While applying traditional control, one needs to know about the model and the objective function formulated in precise terms. This makes it very difficult to apply in many cases.

- By applying fuzzy logic for control we can utilize the human expertise and experience for designing a controller.

- The fuzzy control rules, basically the IF-THEN rules, can be best utilized in designing a controller.

3.1) Architecture of Fuzzy Logic Control

Fuzzy logic works on the concept of deciding the output based on assumptions. It works based on sets. Each set represents some linguistic variables defining the possible state of the output. Each possible state of the input and the degrees of change of the state are a part of the set, depending upon which the output is predicted. It works on the principle of If-else-the, i.e. If A AND B Then Z.
Suppose we want to control a system where the output can be anywhere in the set X, with a generic value x, such that x belongs to X. Consider a particular set A which is a subset of X such that all members of A belong to the interval 0 and 1. The set A is known as a fuzzy set and the value of fA(x) at x denotes the degree of membership of x in that set. The output is decided based on the degree of membership of x in the set. This assigning of membership depends on the assumption of the outputs depending on the inputs and the rate of change of the inputs.
These fuzzy sets are represented graphically using membership functions and the output is decided based on the degree of membership in each part of the function. The membership of the sets is decided by the IF-Else logic.
Generally, the variables of the set are the state of the inputs and the degrees of changes of the input and the membership of the output depends on the logic of AND operation of the state of the input and the rate of change of the input. For a multi-input system, the variables can also be the different inputs and the output can be the possible result of the AND operation between the variables.

The following block diagram represents the architecture of a Fuzzy Logic Control (FLC).

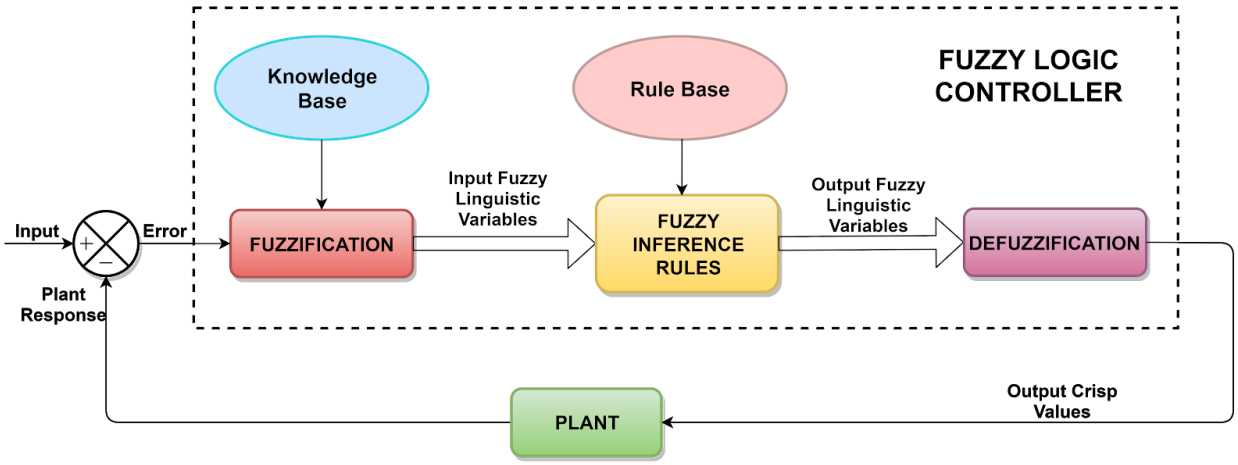


Fig. 7. Block Diagram Representation of a Fuzzy Logic Controller

Followings are the major components of the FLC as shown in the above figure −

- **Fuzzifier** − The role of fuzzifier is to convert the crisp input values into fuzzy values. **Fuzzification** is the method of transforming a crisp quantity into a fuzzy quantity. This can be achieved by identifying the various known crisp and deterministic quantities as completely nondeterministic and quite uncertain in nature. This uncertainty may have emerged because of vagueness and imprecision which then lead the variables to be represented by a membership function as they can be fuzzy in nature.

- **Fuzzy Knowledge Base** − It stores the knowledge about all the input-output fuzzy relationships. It also has the membership function which defines the input variables to the fuzzy rule base and the output variables to the plant under control.

- **Fuzzy Rule Base** − It stores the knowledge about the operation of the process of domain. These are the If-Else Rules which form the relation between the fuzzified inputs and the outputs. The output of a FLC is decided on the basis of these Fuzzy Rules.

- **Inference Engine** − It acts as a kernel of any FLC. Basically it simulates human decisions by performing approximate reasoning.

- **Defuzzifier** − The role of a defuzzifier is to convert the fuzzy values into crisp values getting from fuzzy inference engine. **Defuzzification** is the inversion of fuzzification, where the mapping is done to convert the crisp results into fuzzy results but here the mapping is done to convert the fuzzy results into crisp results. This process is capable of generating a non fuzzy control action which illustrates the possibility distribution of an inferred fuzzy control action. Defuzzification process can also be treated as the rounding off process, where a

fuzzy set having a group of membership values on the unit interval is reduced to a single scalar quantity.

3.2) Advantages of Fuzzy Logic Control

Following are the advantages of Fuzzy Logic Control.

- **Cheaper** − Developing a FLC is comparatively cheaper than developing model based or other controller in terms of performance.
- **Robust** − FLCs are more robust than PID controllers because of their capability to cover a huge range of operating conditions.
- **Customizable** − FLCs are customizable.
- **Emulate human deductive thinking** − Basically FLC is designed to emulate human deductive thinking, the process people use to infer conclusion from what they know.
- **Reliability** − FLC is more reliable than conventional control system.
- **Efficiency** − Fuzzy logic provides more efficiency when applied in control system.

3.3) Adaptive Fuzzy PID Controller

The mathematics in a PID control equation is complex with multiple variables and constants interacting. In any given application these are selected to follow the target value as closely as possible, within the constraints imposed by the process itself and the instrumentation.
Three issues common to almost every process control application are:

- Time delays or lag
- Step function response
- “Ramp & Soak” function response

In many situations the output can take a long, and perhaps also variable, time to react to input changes. When the target value changes instantaneously PID forces the system to apply a large correcting factor, which again can lead to overshoot. Alternatively, the system may become saturated, unable to supply sufficient correction, adding to the impact of the “I” term. Tracking a gradual change in setpoint can challenge PID control systems.
As a result, selecting the optimal values is something of a trial-and-error process known as “tuning.” Over the years many approaches to tuning have been developed, the most satisfactory of which appears to be the “Ziegler Nichols” method. However, this produces high levels of oscillation, which can be problematic in some situations. Tuning of PID loops depends on heuristics yet often ends up being sub-optimal. Fuzzy logic provides an alternative to approaches such as Ziegler Nichols, and a growing body of research suggests it yields superior results. Thus it would seem an ideal way to control many complex processes is with a PID controller tuned with fuzzy logic.

As long as the irregularity lies in that dimension in which fuzzy decisions are being based or associated, the result should be enhanced performance. This enhanced performance should be demonstrated in both the transient and steady state response.

If the system tends to have changing properties or some irregularities, a Fuzzy Logic controller should offer a better alternative to the constant adjustment of PID parameters. The following figure shows a block diagram representation of the adaptive Fuzzy - PID controller that can be used in controlling the speed of a DC motor.

The adaptive controller comprises two segments: conventional PID and FLC. Kp, Ki and Kd are the preliminary values of parameters of PID controller, Kp1, Ki1 and Kd1 are the gain outputs of FLC. Then, the tuned parameters will be given by:

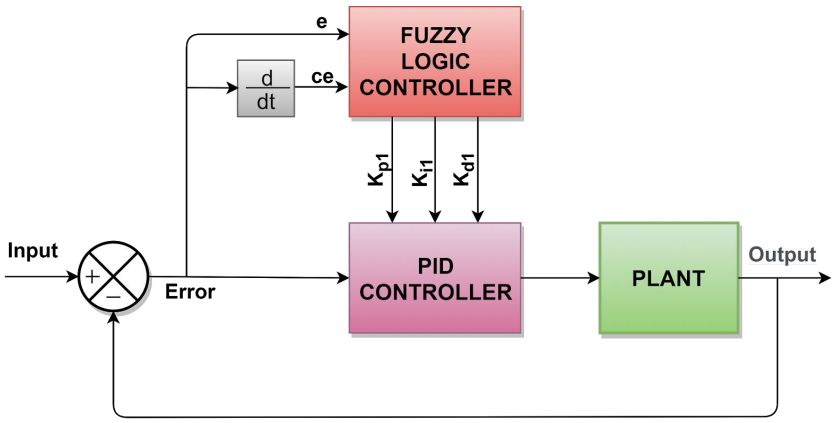


Fig. 8. Block Diagram Representation of an adaptive Fuzzy - PID Controller

$$K_{p2} = K_p . K_{p1} \quad (11)$$

$$K_{i2} = K_i . K_{i1} \quad (12)$$

$$K_{d2} = K_d . K_{d1} \quad (13)$$

Now the output control action of PID controller after self tuning will be given as:

$$U^{PID} = K_{p2} e(t) + K_{i2} \int e(t) + K_{d2} \frac{de(t)}{dt} \quad (14)$$

4. DESIGN

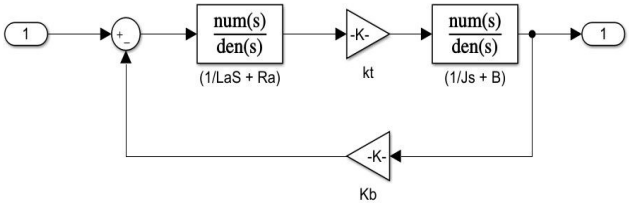
4.1) Modelling The DC Motor

Using the block diagram representation of an Armature controlled DC motor discussed previously, the DC motor was modelled in simulink through a sub-system block using actual parameters from a real DC motor given below

Table 1. DC Motor Parameters

Parameter	Nomenclature	Value
Moment of Inertia	J	0.093 $Kg.m^2$
Friction Coefficient	B	0.008 $N.ms$
Back emf constant	K_b	0.6 $V/rad.s^{-1}$
Torque constant	K_t	0.7274 Nm/A
Armature resistance	R_a	0.6 ohm
Armatur Inductance	L_a	0.006 H

The simulink Model and the Complete transfer function of the motor are as follows:

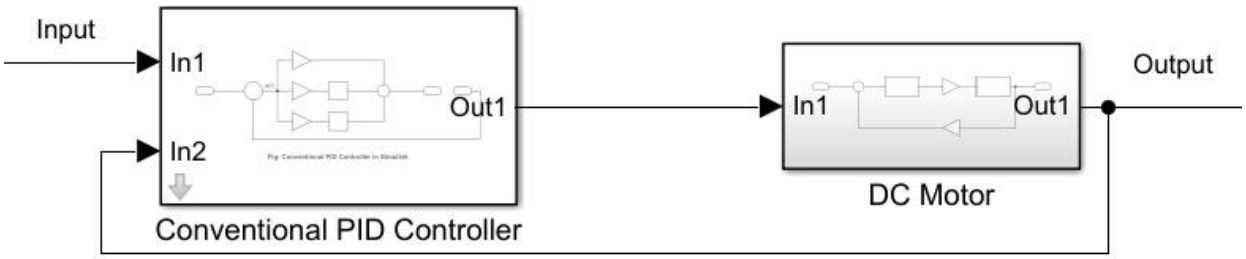



```
P =

                0.7274
  ---------------------------------
  0.000558 s^2 + 0.05585 s + 0.4412

Continuous-time transfer function.
```

Fig. 9. Model of DC Motor in Simulink

4.2) Designing The PID Controller

Firstly, the conventional PID controller was designed for analyzing the performance of conventional PID controller on the dc motor speed control.



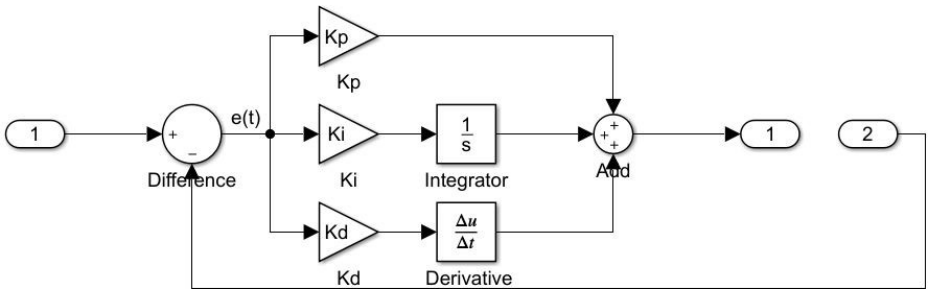


Fig. 10. Conventional PID Controller in Simulink

4.3) Tuning The PID Parameters

The tuning of PID parameters was done using the PID Tuner toolbox available in MATLAB. To use the PID Tuner a Plant has to be identified or defined for the system. The following MATLAB script was used to define a Plant function and a Baseline PID Controller with random PID Parameters.

```
s = tf('s');
tf1 = 1/(0.006*s + 0.6);
tf2 = 1/(0.093*s + 0.008);
kt = 0.7274;
kb = 0.6;
DCMotor = feedback(tf1*kt*tf2, kb); %Defining the Plant (DC Motor)
% Setting random parameters for Baseline
% Determined using hit and trial on
% the system
Kp = 0.1;
Ki = 10;
Kd = 0.01;
C = pid(Kp,Ki,Kd); %modelling a PID controller
pidTuner(DCMotor,C); % Starting the PID Tuner
```

Using the defined Plant function, the PID tuner creates another PID control and the response of the baseline controller and the new created PID controller is shown. After that the PID parameters are tuned by changing the Robustness and Response time of the system.

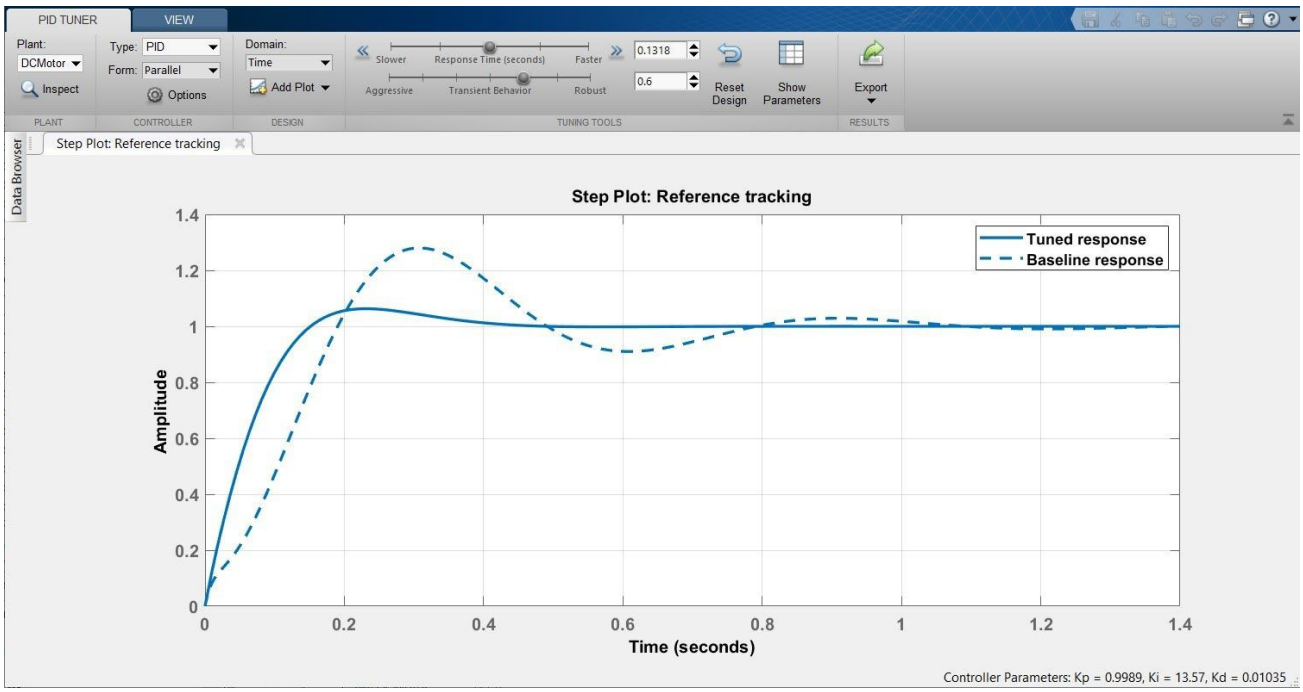


Fig. 11. Tuning the PID parameters in PID Tuner

Controller Parameters

	Tuned	Baseline
Kp	0.99889	0.1
Ki	13.5694	10
Kd	0.010346	0.01
Tf	n/a	n/a

Performance and Robustness

	Tuned	Baseline
Rise time	0.108 seconds	0.157 seconds
Settling time	0.371 seconds	0.989 seconds
Overshoot	6.3 %	28 %
Peak	1.06	1.28
Gain margin	Inf dB @ NaN rad/s	Inf dB @ NaN rad/s
Phase margin	73.8 deg @ 15.2 rad/s	41.4 deg @ 9.83 rad/s
Closed-loop stability	Stable	Stable

Fig. 12. Tuned Parameters

A comparison between the performance of the baseline controller and the tuned controller is also given by the PID Tuner. The Table shows the various Control parameters such as the PID Gains, rise time, settling time, Peak Overshoot etc.

Based on the Tuned Parameters both the conventional and the adaptive Fuzzy PID were designed.

4.4) Designing The Fuzzy Logic Controller

The fuzzy logic controller was designed using the fuzzy logic toolbox in MATLAB.

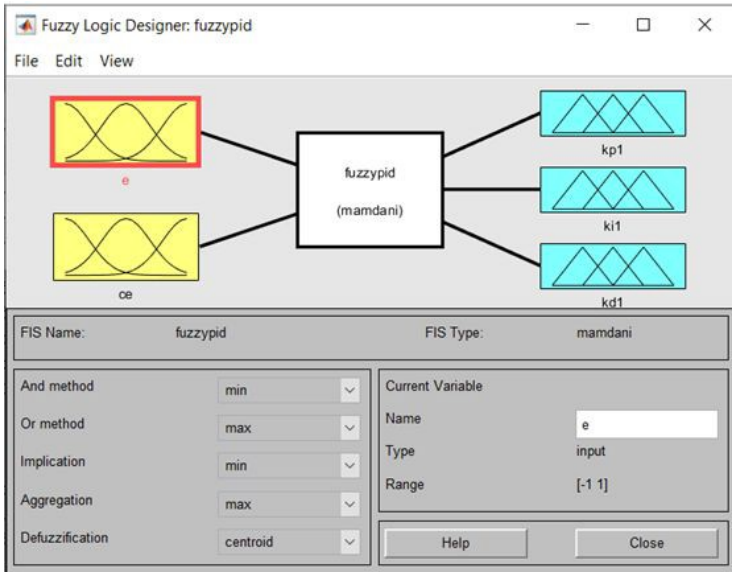


Fig. 13. Fuzzy Logic Designer

In fuzzy logic controller, the inputs are speed error (e) and change in speed error (ce) and the outputs are Kp1, Ki1 and Kd1.

PARAMETER	RANGE
e, ce	-1 to 1
Kp1, Ki1, Kd1	0 to 2

Seven Membership functions representing different value ranges were created for each of the inputs and outputs.

Membership Functions for Inputs e and ce

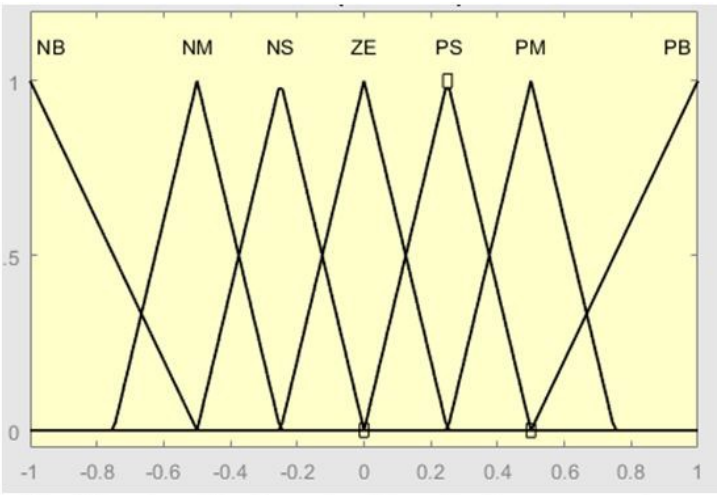


Fig. 14. Membership Functions Plot for e and ce

Table 2. Abbreviations

NB	NEGATIVE BIG
NM	NEGATIVE MEDIUM
NS	NEGATIVE SMALL
ZE	ZERO
PS	POSITIVE SMALL
PM	POSITIVE MEDIUM
PB	POSITIVE BIG

The seven membership functions ranging from negative big to positive big define the fuzzy ranges for the input variables e and ce. The inputs are normalized before sending to the fuzzy logic controller. The normalization helps in defining the clear boundary of the range of the input.

Membership Functions for Output Kp1, Ki1, Kd1

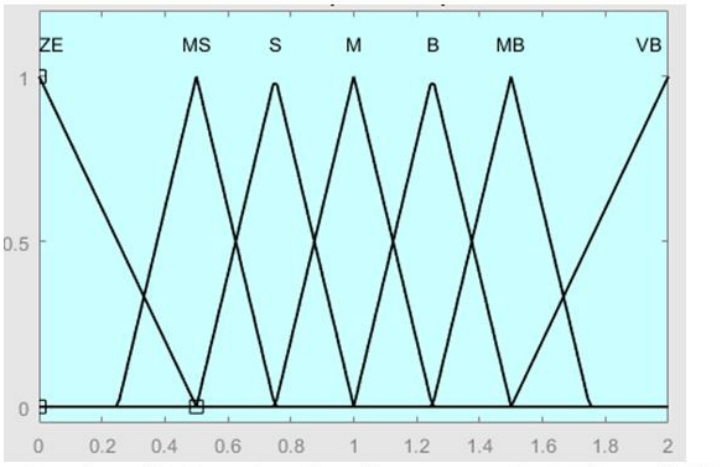


Fig. 15. Membership Functions Plot for Kp1, Ki1, and Kd1

Table 3. Abbreviations

ZE	ZERO
MS	MUCH SMALL
S	SMALL
M	MEDIUM
B	BIG
MB	MUCH BIG
VB	VERY BIG

The seven membership functions ranging from zero to very big define the fuzzy ranges for the outputs that are the Fuzzy Logic Gains. The outputs range from 0 to 2 which provides sufficient adaptive tuning during the working of the system.

4.5) Rule Base for the Fuzzy Logic Controller

The rule base is formed using the inputs and the corresponding outputs, i.e. what should be the outputs for a particular set of inputs. These relations are made on the basis of the working and significance of the three PID Gains.

Table 4. Relation between PID gains and System response

Parameter increase	Rise time	Overshoot	Settling time	Steady state error
K_p	Decreases	Increases	Small change	Decreases
K_i	Decreases	Increases	Increases	Highly reduced
K_d	Small change	Decreases	Decreases	Small change

The table shows the relation between the change in the PID gains and its respective impact on the output.

Seven membership functions are considered for inputs and outputs thus it will give total 49 rules, but for simplicity we consider only five membership functions thus producing only 25 rules.

Table 5. Fuzzy Rules for Kp1

ce/e	NB	NS	ZE	PS	PB
NB	VB	VB	VB	VB	VB
NS	B	B	B	MB	VB
ZE	ZE	ZE	MS	S	S
PS	B	B	B	MB	VB
PB	VB	VB	VB	VB	VB

Table 6. Fuzzy Rules for Ki1

ce/e	NB	NS	ZE	PS	PB
NB	M	M	M	M	M
NS	S	S	S	S	S
ZE	MS	MS	ZE	MS	MS
PS	S	S	S	S	S
PB	M	M	M	M	M

Table 7. Fuzzy Rules for Kd1

ce/e	NB	NS	ZE	PS	PB
NB	ZE	S	M	MB	VB
NS	S	B	MB	VB	VB
ZE	M	MB	MB	VB	VB
PS	B	VB	VB	VB	VB
PB	VB	VB	VB	VB	VB

The syntax for defining rules in rule base is:

$$If\ \left(e\ is\ NB\right)\ and\ \left(ce\ is\ NB\right)\ then$$
$$\left(K_p\ is\ VB\right)\ \left(K_i\ is\ M\right)\ \left(K_d\ is\ ZE\right)$$

4.6) Adaptive Fuzzy - PID Controller

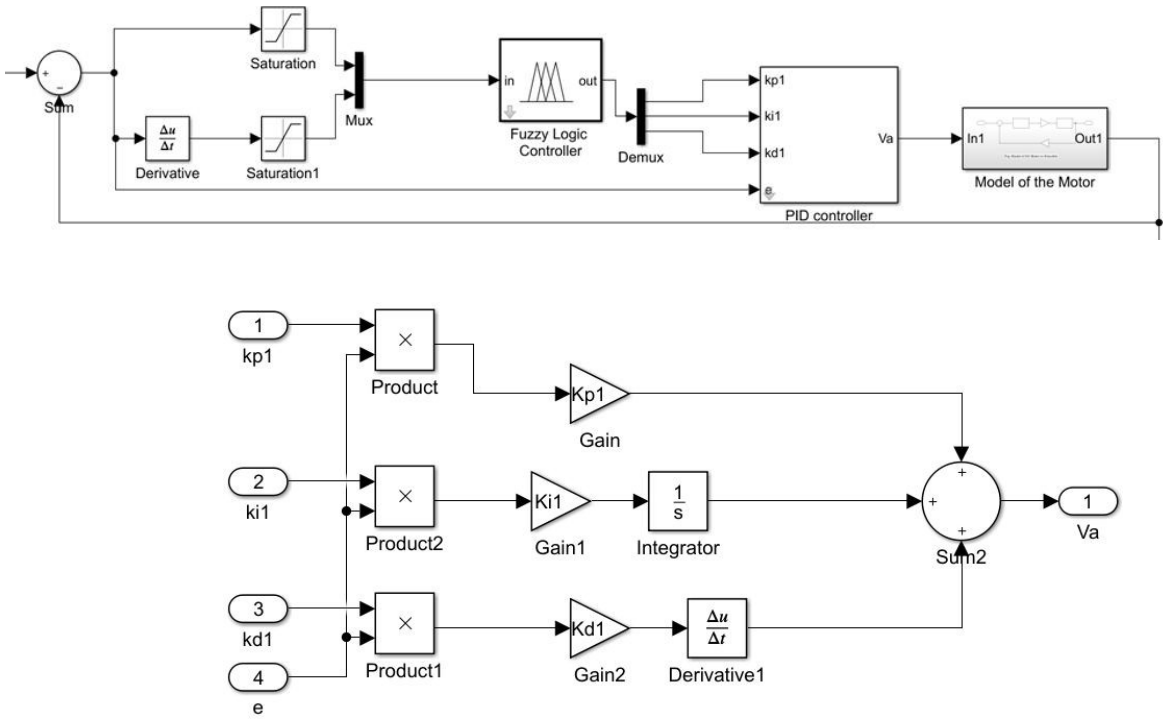


Fig. 16. Adaptive PID Controller in Simulink

The adaptive PID Controller was created as a subsystem and the inputs were masked to create a single block for changing the parameters.

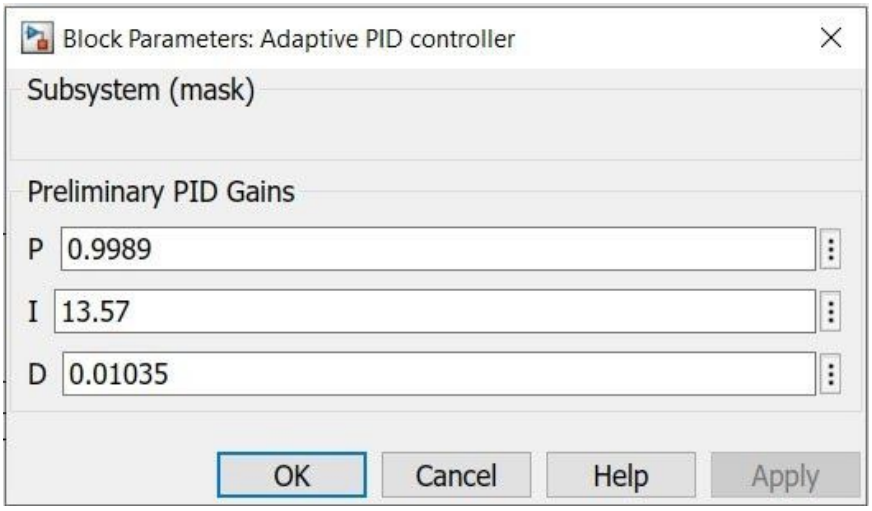


Fig. 17. Masked Inputs for Adaptive PID Controller Block

5. SIMULATION AND RESULTS

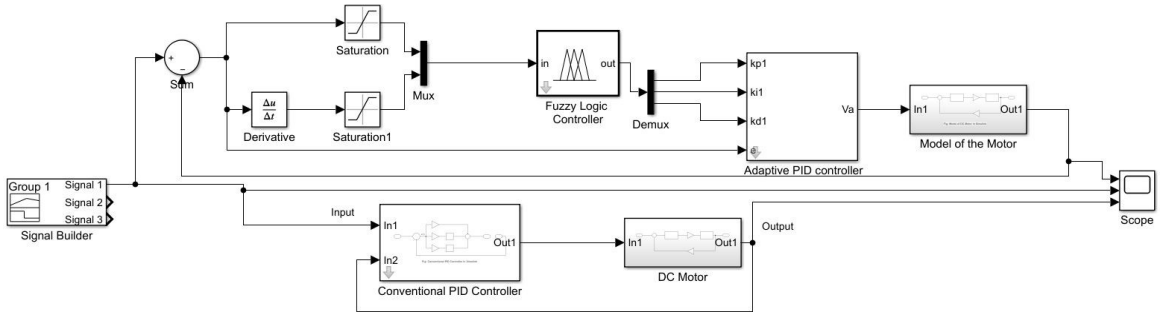
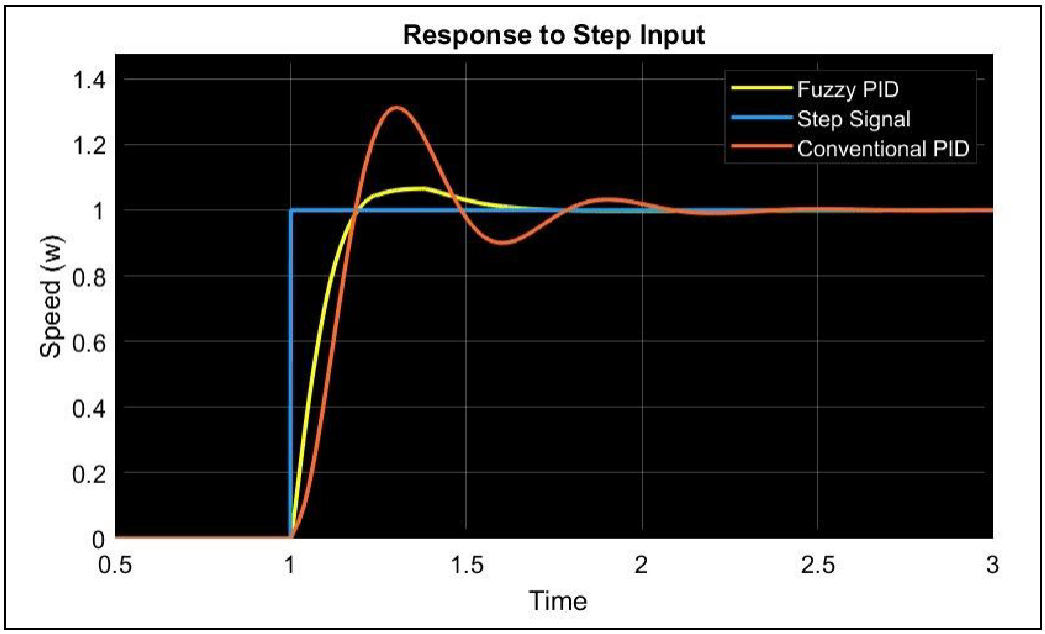


Fig. 18. Complete System in Simulink

The Signal Builder Block was used to input different types of signals to compare the performance of conventional PID Controller and the adaptive Fuzzy-PID Controller. For the comparison, the same parameters were used in both the controllers for every simulation.

Response on Step Input with untuned Parameters



Kp = 0.1
Ki = 10
Kd = 0.01

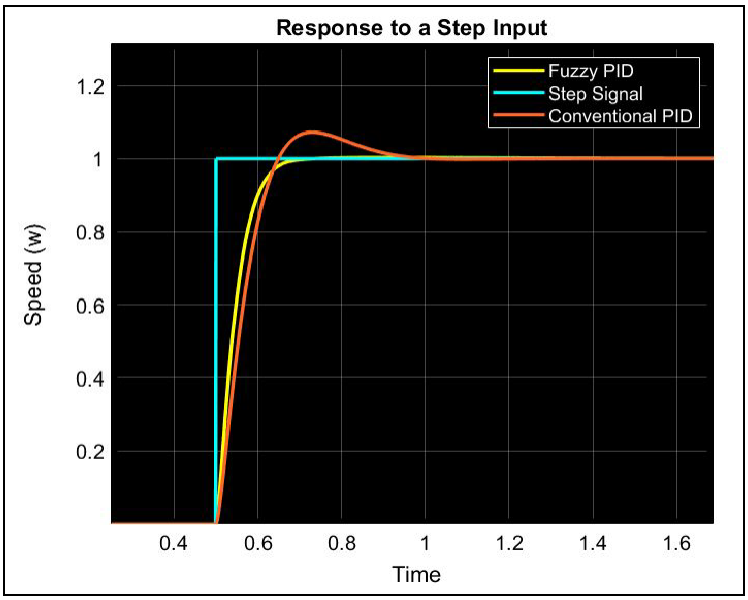
Clearly, with untuned parameters Adaptive Fuzzy PID performed better as compared to conventional PID in terms of Peak Overshoot, Settling time and rise time.

Fig. 19. Response to custom input with untuned parameters

Table 8. System Characteristics for Untuned Parameters

Controller Type	System Characteristics		
	Rise Time (ms)	Overshoot (%)	Settling Time(s)
PID	138.56	30.921	0.993
Fuzzy PID	130.72	6.98	0.546

Response on Step Input with Tuned Parameters



Kp = 0.9989
Ki = 13.57
Kd = 0.01035

When Tuned Parameters are used the performance of Fuzzy PID is much better than the Conventional PID. When step is applied the Kp gain is increased by the correction factors and this gives a smaller rise time. Adaptive changes in the Ki and Kd helps in achieving extremely small settling time.

Fig. 20. Response to a Step Input

Response on Custom Input with Tuned Parameters

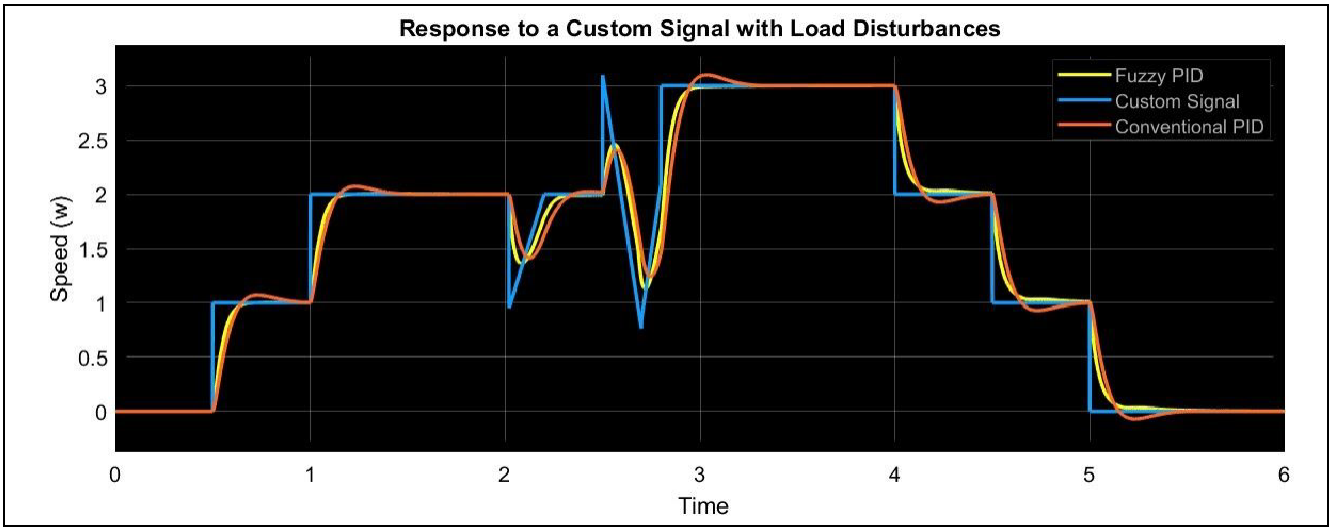


Fig. 21. Response to Custom input

When a load is applied to the system between t=2 to t=3 seconds, the Fuzzy PID responds and recovers faster as compared to the convention PID controller.

Table 9. Performance Comparison of PID with Adaptive Fuzzy- PID

Controller Type	System Characteristics		
	Rise Time (ms)	Overshoot (%)	Settling Time(s)
PID	102.669	6.989	0.387
Fuzzy PID	88.22	0.5	0.120

6. CONCLUSION

In this project, two techniques, PID and based adaptive fuzzy PID controller are presented for speed control of motor and simulated using MATLAB/Simulink. In adaptive fuzzy PID controller, the controller coefficients are continuously updated according to error and change in error using fuzzy rules. The controller response is tested for reference speed tracking and load changes. During load disturbances, the adaptive fuzzy PID controller adapts the controller parameters to maintain desired operation which shows robustness of the system. The two control techniques are compared in terms of rise time, overshoot and settling time and the comparison shows that the PID controller suffers from high overshoot and high-rise time whereas in case of adaptive fuzzy PID controller the overshoot, rise time and settling time are reduced.

REFERENCES

1. M. A. Shamseldin and A. A. E.-. Samahy, "Speed control of BLDC motor by using PID control and self-tuning fuzzy PID controller," in 15thInternational Workshop on Research and Education in Mechatronics (REM), 2014, pp. 1-9.

2. W. Gubara, M. Elnaim, and S. F. Babiker, "Comparative study on the speed of DC motors using PID and FLC," 2016 Conference of Basic Sciences and Engineering Studies (SGCAC), 2016, pp. 24-29.

3. KiamHeong, G. Chong, and L. Yun, "PID control system analysis,design, and technology," IEEE Transactions on Control Systems Technology, vol. 13, pp. 559-576, 2005.

4. R. Kandiban and R. Arul Mozhiyil, "Speed Control of BLDC Motor Using Adaptive Fuzzy PID Controller," Procedia Engineering, vol. 38, pp. 306-313, 2012/01/01 2012.

5. M. Kushwah and A. Patra, "Tuning PID controller for speed control of dc motor using soft computing techniques-a review," Advance in Electronic and Electric Engineering, vol. 4, pp. 141-148, 2014.

6. P. H. Krishnan and M. Arjun, "Control of BLDC motor based on adaptive fuzzy logic PID controller," 2014 International Conference on Green Computing Communication and Electrical Engineering (ICGCCEE),2014, pp. 1-5.

7. A. El-samahy and M. A. Shamseldin, "Brushless DC motor tracking control using self-tuning fuzzy PID control and model reference adaptive control," Ain Shams Engineering Journal.

8. H. Ahmed and A. Rajoriya, "Performance Assessment of Tuning Methods for PID Controller Parameter used for Position Control of DC Motor, "International Journal of u-and e-Service, Science and Technology, vol. 7,pp. 139-150, 2014.

9. Munadi and M. A. Akbar, "Simulation of fuzzy logic control for DC servo motor using Arduino based on MATLAB/Simulink," in 2014