Abstract—Brush less DC motors are widely used for many industrial applications because of their high efficiency, high torque and compact size. The proposed speed controlling technique used to control the speed of a brushless DC motor is based on proportional integral derivative controller and Fuzzy proportional integral derivative controller. This paper describes the overview of performance of conventional proportional integral derivative controller and Fuzzy proportional integral derivative controller. It is difficult to tune the parameters and get satisfied control characteristics by using normal conventional proportional integral derivative controller. As the Fuzzy has the ability to satisfy control characteristics and it is easy for computing in order to control the Brushless DC motor. Fuzzy proportional integral derivative controller is designed to control the speed of Brushless DC motor. The modeling, control and simulation of the Brush less DC motor have been done using the software package MATLAB to control the speed of Brushless DC motor. The results proved that a Fuzzy proportional integral derivative controller is better than Conventional proportional integral derivative controller.

Keywords: Fuzzy Proportional Integral Derivative and Brushless DC Motor, Speed Control.

I. INTRODUCTION
In most of the industrial processes with different degrees of nonlinear, parameter variability and uncertainty of mathematical model of the system, tuning PID control parameters is very difficult. Therefore, it's difficult to achieve the optimal state under field conditions in the actual production. Fuzzy PID control method is a better method of controlling, to the complex and unclear model systems, it can give simple and effective control, good dynamic response, rising time, overstrike characteristics.

The aim of this paper is that it shows the dynamic response of speed with design of conventional PID controller and Fuzzy PID controller to control the speed of motor for keeping the motor speed to be constant when the load varies. It shows that the performance of Fuzzy PID controller is better than conventional PID controller.

II. BRUSH LESS DC MOTOR
2.1 Introduction
The PMBLDC motor is becoming popular in various applications because of its high efficiency, high power factor, high torque, simple control and lower maintenance. The major disadvantage with permanent magnet motors is their higher cost and relatively higher complexity introduced by the power electronic converter used to drive them. The added complexity is evident in the development of a torque/speed regulator.

As the name implies, BLDC motors do not use brushes for commutation; instead, they are electronically commutated. BLDC motors have many advantages over brushed DC motors and induction motors. A few of these are:
- Better speed versus torque characteristics
- High dynamic response
- High efficiency
- Long operating life
- Noiseless operation
- Higher speed ranges

In addition, the ratio of torque delivered to the size of the motor is higher, making it useful in applications where space and weight are critical factors. In this application note, we will discuss in detail the construction, working principle, characteristics and typical applications of BLDC motors.

2.2 Basic Structure of Brush Less DC Motor
The construction of modern brushless motors is very similar to the ac motor, known as the permanent magnet
synchronous motor. Fig 2.1 illustrates the structure of a typical three-phase brushless dc motor. The stator windings are similar to those in a poly phase ac motor, and the rotor is composed of one or more permanent magnets. Brushless dc motors are different from ac synchronous motors in that the former incorporates some means to detect the rotor position (or magnetic poles) to produce signals to control the electronic switches as shown in Fig 2.2. The most common position/pole sensor is the Hall element, but some motors use optical sensors.

![Fig. 2.1 Disassembled view of a Brush Less DC Motor](image)

**Fig. 2.1 Disassembled view of a Brush Less DC Motor**

**Fig. 2.2 Brushless dc motor = Permanent magnet ac motor + Electronic commutator**

### 2.3 Modeling of Brush Less DC Motor Drive System

The BLDC motor drive system is modeled based on the assumptions that all the stator phase windings have equal resistance per phase; constant self and mutual inductances; power semiconductor devices are ideal; iron losses are negligible; and the motor is unsaturated. The equivalent circuit of the BLDC motor drive system is shown in Fig 2.3.

![Fig 2.3 Equivalent circuit of the BLDC motor drive system](image)

**Fig 2.3 Equivalent circuit of the BLDC motor drive system.**

The line to line voltage equations are expressed in matrix form as

\[
\begin{bmatrix}
V_{ab} \\
V_{bc} \\
V_{ca}
\end{bmatrix} = \begin{bmatrix}
R & -R & 0 \\
0 & R & -R \\
-R & 0 & R
\end{bmatrix} \begin{bmatrix}
i_a \\
i_b \\
i_c
\end{bmatrix} + \begin{bmatrix}
L \cdot M & M \cdot L & 0 \\
0 & L \cdot M & M \cdot L \\
M \cdot L & 0 & L \cdot M
\end{bmatrix} \begin{bmatrix}
e_a - e_b \\
e_b - e_c \\
e_c - e_a
\end{bmatrix}
\]

Since the mutual inductance is negligible as compared to the self-inductance, the aforementioned matrix equation can be rewritten as:

\[
\begin{bmatrix}
V_{ab} \\
V_{bc} \\
V_{ca}
\end{bmatrix} = \begin{bmatrix}
R & -R & 0 \\
0 & R & -R \\
-R & 0 & R
\end{bmatrix} \begin{bmatrix}
i_a \\
i_b \\
i_c
\end{bmatrix} + \begin{bmatrix}
L & 0 & 0 \\
0 & L & 0 \\
0 & 0 & L
\end{bmatrix} \begin{bmatrix}
e_a \\
e_b \\
e_c
\end{bmatrix} - \begin{bmatrix}
e_a \\
e_b \\
e_c
\end{bmatrix}
\]

Where L and M are self-inductance and mutual inductance per phase; R is the stator winding resistance per phase; \(e_a\), \(e_b\), and \(e_c\) are the back EMFs of phases a, b, and c, respectively; \(i_a\), \(i_b\), and \(i_c\) are the phase currents of phases a, b, and c respectively.

### III. PROPORTIONAL INTEGRAL DERIVATIVE DESIGN (PID)

#### 3.1 Introduction

The control of electrical motors used in high-performance servo drives and robots, control concepts are used to achieve high dynamic performance. PID controllers are extensively used in servo control system. The performance of PID controllers is sensitive to system parameter variations. Servomotors are used in many automatic systems, including drives for printers, tape recorders, and robotic manipulators.

#### 3.2 PID Controllers

The PID algorithm is the most popular feedback controller used within the process industries. It is a robust easy to understand that can provide excellent control performance despite the varied dynamic characteristics of process plant. Considering the characteristics parameters-proportional(P), integral(I) and derivative(D), controls as applied to the diagram below in figure 3.1, the system, S is to be controlled using the controller, C; where controller efficiency depends on the P, I, and D parameters.

![Fig 3.1 A typical system with a controller](image)

The controller provides the excitation needed by the system and it is designed to control the overall behavior of the system. The PID controller has several categories of structural arrangements. The most common of these are the series and parallel structures and in some cases, these are the hybrid form of the series and parallel structures.

Typically the function of the form shown in equation below is applicable in this kind of PID controller design.
\[ K_P + \frac{K_I}{s} + K_D \cdot s = \frac{K_P s^2 + K_D s + K_I}{s} \] 

Where,
- \( K_P \): Proportional gain
- \( K_I \): Integral gain
- \( K_D \): Derivative gain

Considering the figure-3.2, variable, \( (e) \) is the sample error, and it is the difference between the desired input value, and the actual output. In a closed loop, \( (e) \) will be sent to the controller, and the controller will perform the integral and derivative computation on the error signal. Thereafter, the signal \( u \), which is the output of the controller is now equal to the sum of \([\text{the product of proportional gain, } K_P \text{ and the magnitude of the error}], [\text{the product of the integral gain, } K_I \text{ and the integral of the error}] \) and \([\text{the product of the derivative gain, } K_D \text{ and the derivative of the error}] \).

That is,
\[ u = K_P e + K_I \int e dt + K_D \frac{de}{dt} \quad -(3.2) \]

The signal value, \( u \) is sent continuously to the plant with every corresponding new output, being obtained as the process continues. The output, is sent back and subsequently new error signal, \( (e) \) is found and the same process repeats itself on and on.

Also, it is very typical to have the PID transfer function written in several forms depending on the arrangement structure. The following equation shows one of the parallel structures:
\[ K_P + \frac{K_I}{s} + K_D \cdot s + K_P s^2 + K_D s + K_I \] 

Where,
- \( K_P \): Proportional gain
- \( T_I \): Integral time or Reset time
- \( T_D \): Derivative time or Rate time

The ability to blend these three parameters will make a very efficient and stable system. It should be noted that the relationship between the three controller parameters may not exactly be accurate because of their interdependency. Therefore, it is very impossible to compute particular parameters which effects would be noticed on the other two.

### IV. FUZZY LOGIC CONTROLLER

#### 4.1 Introduction

In recent years, the number and variety of applications of Fuzzy Logic (FL) have increased significantly. The applications range from consumer products such as cameras, camcorders, washing machines, and microwave ovens to industrial process control, medical instrumentation, decision-support systems, and portfolio selection. To understand why use of Fuzzy Logic has grown, it must be first understood as what is meant by Fuzzy Logic.

#### 4.2 The Fuzzy Logic Concept

Fuzzy logic arose from a desire to incorporate logical reasoning and the intuitive decision making of an expert operator into an automated system. The aim is to make decisions based on a number of learned or predefined rules, rather than numerical calculations. Fuzzy logic incorporates a rule-base structure in attempting to make decisions.

In fuzzy logic control, the term “linguistic variable” refers to whatever state variables the system designer is interested in. Linguistic variables that are often used in control applications include Speed, Speed Error, Position, and Derivative of Position Error. The fuzzy variable is perhaps better described as a fuzzy linguistic qualifier. Thus the fuzzy qualifier performs classification (qualification) of the linguistic variables. The fuzzy variables frequently employed include Negative Large, Positive Small and Zero. Several papers in the literature use the term “fuzzy set” instead of “fuzzy variable”, however; the concept remains the same. Once the linguistic and fuzzy variables have been specified, the complete inference system can be defined. The fuzzy linguistic universe, \( U \), is defined as the collection of all the fuzzy variables used to describe the linguistic variables, i.e. the set \( U \) for a particular system could be comprised of Negative Small (NS), Zero (ZE) and Positive Small (PS). Thus, in this case the set \( U \) is equal to the set of [NS, ZE, PS].

### Table 3.1 - PID controller parameter characteristics on a typical system

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Rise Time</th>
<th>Over Shoot</th>
<th>Settling Time</th>
<th>Steady-State Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>( K_P )</td>
<td>Decreases</td>
<td>Increases</td>
<td>Small change</td>
<td>Decreases</td>
</tr>
<tr>
<td>( K_I )</td>
<td>Decreases</td>
<td>Increases</td>
<td>Increases</td>
<td>Eliminates</td>
</tr>
<tr>
<td>( K_D )</td>
<td>Small change</td>
<td>Decreases</td>
<td>Decreases</td>
<td>Small change</td>
</tr>
</tbody>
</table>

Sriramoju et al., Comparison of Fuzzy PID Controller with Conventional PID Controller in Controlling the Speed of a Brushless DC Motor

International Electrical Engineering Journal (IEEJ)
Vol. 5 (2014) No.12, pp. 1665-1672
ISSN 2078-2365
http://www.ieejournal.com/
4.2.1 The Fuzzy Inference System (FIS)

The basic fuzzy inference system (FIS) can be classified as:
- Type 1 Fuzzy Input Fuzzy Output (FIFO)
- Type 2 Fuzzy Input Crisp Output (FICO)

Type 2 differs from the first in that the crisp output values are predefined and, thus, built into the inference engine of the FIS. In contrast, type 1 produces linguistic outputs. Type 1 is more general than type 2 as it allows redefinition of the response without having to redesign the entire inference engine. One drawback is the additional step required, converting the fuzzy output of the FIS to a crisp output. Developing a FIS and applying it to a control problem involves several steps:
1. Fuzzification
2. Fuzzy rule evaluation (fuzzy inference engine)
3. Defuzzification.

The total fuzzy inference system is a mechanism that relates the inputs to a specific output or set of outputs. First, the inputs are categorized linguistically (fuzzification), then the linguistic inputs are related to outputs (fuzzy inference) and, finally, all the different outputs are combined to produce a single output (defuzzification). Figure 4.1 shows a block diagram of the fuzzy inference system.

4.3 Fuzzy Logic Toolbox

The Fuzzy Logic Toolbox extends the MATLAB technical computing environment with tools for designing systems based on fuzzy logic. Graphical User Interfaces (GUIs) guides through the steps of fuzzy inference system design. Functions are provided for many common fuzzy logic methods, including Fuzzy Clustering and Adaptive Neuro Fuzzy learning.

The toolbox allows to model complex system behaviors using simple logic rules and then implements these rules in a fuzzy inference system. It can be used as a standalone fuzzy inference engine. Alternatively, fuzzy inference blocks in simulink can be used and simulate the fuzzy systems within a comprehensive model of the entire dynamic system.

4.4 Building a Fuzzy Inference System

Fuzzy inference is a method that interprets the values in the input vector and, based on user defined rules, assigns values to the output vector. Using the GUI editors and viewers in the Fuzzy Logic Toolbox, building the rules set, define the membership functions, and analyze the behavior of a Fuzzy Inference System (FIS).

Although it’s possible to use the Fuzzy Logic Toolbox by working strictly from the command line, in general it’s much easier to build a system graphically. There are five primary GUI tools for building, editing, and observing fuzzy inference systems in the Fuzzy Logic Toolbox. These GUIs are dynamically linked, in that changes you make to the FIS using one of them, can affect what you see on any of the other open GUIs. You can have any or all of them open for any given system.

4.5 The Membership Function Editor

The Membership Function Editor shares some features with the FIS Editor. In fact, all of the five basic GUI tools have similar menu options, status lines, and Help and Close buttons. The Membership Function Editor is the tool that displays and edits all of the membership functions associated with all of the input and output variables for the entire fuzzy inference system. The five primary GUI tools for building, editing, and observing fuzzy inference systems in the Fuzzy Logic Toolbox are dynamically linked, in that changes you make to the FIS using one of them, can affect what you see on any of the other open GUIs. You can have any or all of them open for any given system. The figure for Membership Function Editor of this paper is shown below:

![Fig. 4.1 Fuzzy inference system](image)

In the Edit pull down menu we can choose add MFs which opens a new window, for selecting the function type and the number of membership functions associated with the selected variable. In the lower right corner of the window are the controls that change the name, type, and parameters (shape), of the membership function, once it has been selected.

The process of specifying the input membership functions for the paper is as follows:
- The input variable, change in error, was selected by double-clicking on it. The Range and the Display Range to the vector was given [-1 1].
- The Add MFs from the Edit menu opens a window pop as shown in Fig.4.3.
The MF type was given trimf and five for Number of MFs.

By clicking once on the curve with the leftmost hump the name of the curve was given NB. To adjust the shape of the membership function, we can use the mouse, as described above, or type in a desired parameter change, and then click on the membership function. The default parameter listing for this curve is [-1.5 -1 -0.4].

The curve with the middle hump was named NS, and the curve with rightmost hump as PS and PB.

The second input variable was given as Error, by clicking on it. Both the Range and the Display Range to the vector was set as [-1 1].

Clicking once directly on the curve with the leftmost triangle, the name of the curve was changed to NB. The default parameter listing for this curve is [-1.5 -1 -0.4].

Curve with the middle hump was named as NS, and the curve with center hump as ZE, the curve with the rightmost hump as PS and PB.

Fig 4.3 Membership function Window

To create the output variable membership functions, the Variable Palette on the left was used, selecting the output variable. The inputs ranged from -1 to 1. Triangular membership function types was selected for the output. First, the Range (and the Display Range) was set to [-1 1], to cover the output range. Initially, the NB membership function will have the parameters [-1.5 -1 -0.4], the ZE membership function will be [-0.4 0 0.4], and the PB membership function will be [0.4 1 1.5]. So, the variables were named and the membership functions were given appropriate shapes and names.

Now to write down the rules to call up the Rule Editor, from View menu. Edit rules was selected.

4.6 The Rule Editor

Constructing rules using the graphical Rule Editor interface is fairly self-evident. Based on the descriptions of the input and output variables defined with the FIS Editor, the Rule Editor allows to construct the rule statements automatically, by clicking on and selecting one item in each input variable box, one item in each output box, and one connection item. Choosing none as one of the variable qualities will exclude that variable from a given rule. Choosing not under any variable name will negate the associated quality.

Rules may be changed, deleted, or added, by clicking on the appropriate button as shown in fig.4.4.

The rule base structure is Mamdani type. The FLC in this paper has two inputs and one output. These are error (E), change in error (CE), and control signal, respectively. A linguistic variable which implies inputs and output have been classified as: NB, NS, ZE, PB, PS. Inputs and output are all normalized in the interval of [-1,1]. It is possible to assign the set of decision rules as shown in table 4.1 below. Each control input has five fuzzy sets so that there are at most 25 fuzzy rules.

Table 4.1 Table of Fuzzy Rules

<table>
<thead>
<tr>
<th>E</th>
<th>CE</th>
<th>NB</th>
<th>NS</th>
<th>ZE</th>
<th>PS</th>
<th>PB</th>
</tr>
</thead>
<tbody>
<tr>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>NS</td>
<td>ZE</td>
<td></td>
</tr>
<tr>
<td>NS</td>
<td>NB</td>
<td>NB</td>
<td>NS</td>
<td>ZE</td>
<td>PS</td>
<td></td>
</tr>
<tr>
<td>ZE</td>
<td>NB</td>
<td>NS</td>
<td>ZE</td>
<td>PS</td>
<td>PB</td>
<td></td>
</tr>
<tr>
<td>PS</td>
<td>NS</td>
<td>ZE</td>
<td>PS</td>
<td>PB</td>
<td>PS</td>
<td></td>
</tr>
<tr>
<td>PB</td>
<td>ZE</td>
<td>PS</td>
<td>PB</td>
<td>PS</td>
<td>ZE</td>
<td></td>
</tr>
</tbody>
</table>

The Rules are formed in the form of If-Then statements as seen from table 4.1 above.

The Rules formed are:
- If error is negative big(NB) and change in error is also negative big(NB), then output is negative big(NB)
If error is negative big(NB) and change in error is negative small(NS), then output is negative big(NB)
If error is negative big(NB) and change in error is zero(ZE), then output is negative big(NB)
If error is negative big(NB) and change in error is positive small(PS), then output is negative small(NS)
If error is negative big(NB) and change in error is also positive big(PS), then output is zero(ZE)
If error is negative small(NS) and change in error is also negative big(NB), then output is negative big(NB)
If error is negative small(NS) and change in error is also zero(ZE), then output is negative big(NB)
If error is negative small(NS) and change in error is also zero(ZE), then output is negative small(NS)
If error is negative small(NS) and change in error is zero(ZE), then output is negative small(NS)
If error is zero(ZE) and change in error is negative big(NB), then output is negative big(NB)
If error is zero(ZE) and change in error is negative small(NS), then output is negative small(NS)
If error is zero(ZE) and change in error is zero(ZE), then output is zero(ZE)
If error is zero(ZE) and change in error is positive small(PS), then output is positive small(PS)
If error is zero(ZE) and change in error is positive big(PB), then output is positive big(PB)
If error is positive small(PS) and change in error is negative big(NB), then output is positive small(PS)
If error is positive big(PB) and change in error is zero(ZE), then output is zero(ZE)
If error is positive big(PB) and change in error is negative big(NB), then output is zero(ZE)
If error is positive big(PB) and change in error is negative small(NS), then output is positive small(PS)
If error is positive big(PB) and change in error is positive small(PS), then output is positive big(PB)
If error is positive big(PB) and change in error is negative big(NB), then output is zero(ZE)
If error is positive big(PB) and change in error is positive small(PS), then output is positive big(PB)
If error is positive big(PB) and change in error is positive big(PB), then output is zero(ZE)

So, the output (voltage) is regulated according to these rules and the speed is controlled.

### 4.7 The Rule Viewer

The Rule Viewer displays a roadmap of the whole fuzzy inference process. It's based on the fuzzy inference diagram described in the previous section. The three small plots across the top of the figure represent the antecedent and consequent of the first rule. Each rule is a row of plots, and each column is a variable.

![Fig 4.5 The rule viewer](image)

The first two columns of plots show the membership functions referenced by the antecedent, or the if-part of each rule. The third column of plots shows the membership functions referenced by the consequent, or the then-part of each rule. Now if we click once on a rule number, the corresponding rule will be displayed at the bottom of the figure. The fourth plot in the third column of plots represents the aggregate weighted decision for the given inference system. This decision will depend on the input values for the
system. The de-fuzzy field output value is shown by the thick line passing through the aggregate fuzzy set.

V. SPEED CONTROL OF BLDC MOTOR

The complete block diagram of speed control of three phase BLDC Motor is shown below in Fig.5.1

![Block Diagram of the BLDC Motor Drive System](image)

5. 1 The Fuzzy PID Controller

In order to achieve optimal tracking performance, the motor speed error ‘E’ and change in error ‘CE’ are used as input linguistic variables to the speed controller, respectively. The controller output is the incremental motor voltage command as shown in Fig.5.2 below

![Simulation Model of Fuzzy PID Controller](image)

The speed is controlled by fuzzy logic controller whose output is Voltage. The linguistic variables of inputs and output have been classified as: NB, NS, ZE, PS, PB. Based on these variables it is possible to assign the set of decision rules as shown in Table4.2 of chapter-4. Twenty five rules have been formed. Inputs and output are all normalized in the interval of [-1, 1].

**Rule 1:** IF speed error is NB and change in error CE is NB, THEN a change in control voltage (output of fuzzy controller) is NB. This rule implies a condition when the measured speed is more than the desired reference speed, that is when overshoot occurs. Accordingly, it requires a large decrease in voltage to bring the measured speed to the desired reference speed.

**Rule 2:** IF speed error is NS and change in error is NB, THEN a change in control voltage (output of fuzzy controller) is NB. This rule implies a condition when the measured speed is little more than the reference speed but the change in error is large.

**Rule 3:** IF speed error is ZE and change in error is NB, THEN a change in control voltage (output of fuzzy controller) is NB. This rule implies a condition when the present measured speed is equal to the reference speed but change in error is large. Accordingly, it requires a large decrease in voltage to bring the measured speed to the desired reference speed.

**Rule 4:** IF speed error is PS and change in error is NB, THEN a change in control voltage (output of fuzzy controller) is NS. This rule implies a condition when the present measured speed is little less than the reference speed but change in error is large. Accordingly, it requires a small decrease in voltage as change in error is more.

**Rule 5:** IF speed error is PB and change in error is NB, THEN a change in control voltage (output of fuzzy controller) is ZE. This rule implies a condition when the present measured speed is very low compared to the reference speed but change in error is large, so positive big and negative big are same value but opposite in sign, accordingly no change in voltage is required.

Similarly, all other rules are followed by the fuzzy logic controller to track the error and maintain constant speed of the motor.

<table>
<thead>
<tr>
<th>Switching Interval in degree</th>
<th>Seq. number</th>
<th>Position sensors</th>
<th>Phase Currents</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 – 60</td>
<td>0</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>60 – 120</td>
<td>1</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>120 – 180</td>
<td>2</td>
<td>0</td>
<td>OFF</td>
</tr>
<tr>
<td>180 – 240</td>
<td>3</td>
<td>0</td>
<td>OFF</td>
</tr>
<tr>
<td>240 – 300</td>
<td>4</td>
<td>0</td>
<td>OFF</td>
</tr>
<tr>
<td>300 – 360</td>
<td>5</td>
<td>1</td>
<td>OFF</td>
</tr>
</tbody>
</table>

VI. RESULTS

6.1 Response of PID and Fuzzy PID controller of BLDC motor with Reference speed of 1500 rpm at no-load.
6.2 Fuzzy PID controller-Motor Back EMF (volts) Vs Time (sec)

![Graph showing Fuzzy PID controller vs conventional PID controller]

Table 6.1: Comparison between PID and Fuzzy PID controllers.

<table>
<thead>
<tr>
<th>Speed (rpm)</th>
<th>PID Controller</th>
<th>Fuzzy PID Controller</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rise Time (t_r)</td>
<td>% Overshoot (M_p)</td>
<td>Settling Time (t_s)</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------</td>
<td>------------------</td>
</tr>
<tr>
<td>1500 no load</td>
<td>0.03</td>
<td>16.53</td>
</tr>
<tr>
<td>1500 Load at start</td>
<td>0.02</td>
<td>18.24</td>
</tr>
<tr>
<td>1500 load impact</td>
<td>0.03</td>
<td>16.82</td>
</tr>
</tbody>
</table>

VI. CONCLUSION

In this paper BLDC motor mathematical model is developed. Finally closed loop speed control BLDC is carried out and simulation results are presented. The performance evaluation results show that this modeling is very useful in studying the high performance drive before taking up the dedicated controller design concept for evaluation of dynamic performance of the motor. It presents simulation results of conventional PID controller and Fuzzy PID controller of three phase BLDC Motor. With results obtained from simulation, it is clear that for the same operation condition the BLDC speed control using Fuzzy PID controller technique had better performance than the conventional PID controller, mainly when the motor was working at lower and higher speeds.

REFERENCES