Scientific Journal of Impact Factor(SJIF): 3.134

e-ISSN(O): 2348-4470
p-ISSN(P): 2348-6406

International Journal of Advance Engineering and Research Development

Volume 2,Issue 5, May -2015

Speed Control of Brushless DC Motor Using Soft Computing Techniques

Abhishek D. Gandhi[1], Suryaprakash Singh[2], Bhautik M. Daxini[3]

[1]*Student, Instrumentation & Control Department, AITS, Rajkot, India*
[2]*Assistant Professor, Electrical Department, AITS, Rajkot, India*
[3]*Assistant Professor, Instrumentation & Control Department, AITS, Rajkot, India*

Abstract*—*Brushless DC (BLDC) Motors are widely used in industries because of their high efficiency and high torque. This paper proposed Conventional and Intelligent controllers to control the speed of BLDC Motor. This paper provides an overview of PID, Fuzzy and ANFIS controller. PID controllers are insufficient to control the speed of BLDC motor as it gives high overshoot in the response. So for the better performance, soft computing techniques such as Fuzzy, Fuzzy+PID and ANFIS are used. The simulation results verify that ANFIS has better control performance than the PID and other soft computing techniques. The modeling, control and effects are studied through computer simulation using MATLAB/SIMULINK toolbox.

Index Terms*—*BLDC, ANFIS, FLC, GGM, PID Controller

I. INTRODUCTION

Conventional DC motors[9] are highly efficient and their characteristics make them suitable for use as servomotors. However, their only drawback is that they need a commutator and brushes which are subject to wear and require maintenance. When the functions of commutator and brushes were implemented by solid-state switches, maintenance-free motors were realized. These motors are now known as brushless dc motors.

Design of the BLDCM drive involves a complex process such as modeling, control scheme selection, simulation and parameters tuning etc. To achieve desired level of performance the motor requires suitable speed controllers.

Speed control of BLDC Motors is generally done using proportional integral Derivative (PID) controller. Conventional PID controllers are widely used in the industry as it has simple control structure and easy to implement but these controllers are insufficient to control the speed of BLDC motor as it gives high overshoot in the response.

Intelligent controllers offer an improvement in the quality of the speed response. Most of these controllers use mathematical models and are sensitive to parametric variations. These controllers are inherently robust to load disturbances.

II. MATHEMATICAL MODELING OF BLDC MOTOR

A. *Transfer Function*

Transfer Function derived by the mathematical modeling[5],

$$G(s) = \frac{1/k_e}{\tau_m \tau_e s^2 + \tau_m s + 1}$$

Where,

τ_m = Mechanical time constant = $\frac{3RJ}{k_e k_t}$

τ_e = Electrical time constant = $\frac{L}{3R}$

R = Terminal resistance phase to phase

L = Terminal inductance phase to phase

J = Rotor inertia

k_e = back emf constant (V-sec/rad)

k_t = torque constant (Nm/A)

B. *Specifications of BLDC Motor*[15]

Parameters	Values
Nominal Voltage	9 V
Terminal Resistance phase to phase	1.25 Ω
Terminal Inductance phase to phase	0.32 mH
Torque Constant	10.4 mNm/A
Mechanical Time Constant	60.5 ms
Rotor Inertia	52.3 gcm^2

C. *Model of BLDC Motor*

Final Transfer Function,

$$G(s) = \frac{32.0816}{5.1624 * 10^{-6} s^2 + 0.0605 s + 1}$$

III. CONTROL SCHEMES

A. *PID Controller*

Proportional-Integral-Derivative (PID)[8] control is the most common control algorithm used in industry and has been universally accepted in industrial control. As the name suggests, PID controller algorithm involves three separate constant parameters and is accordingly sometimes called three-term control: proportional, integral and derivative.

The main aim of the PID controller is to sense the sensor signal and evaluate the desired output by calculating the proportional, integral and derivative responses and summing those three components to generate the output. Most of the time, system is affected not only by the actuator output but also by the external factors which are called as the disturbances. PID controller is usually designed to eliminate the effects of the disturbances. Figure 1 shows the typical block diagram of the PID controller.

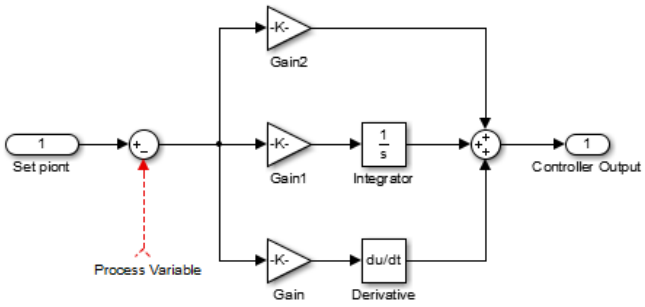

Fig. 1 : PID Block Diagram

Closed Loop Response :

Parameters	Rise Time	Overshoot	Settling Time	S-S Error
K_P	Decrease	Increase	Small Change	Decrease
K_I	Decrease	Increase	Increase	Eliminate
K_D	Small Change	Decrease	Decrease	No Change

PID Tuning :

There are various method for PID tuning, here two method are used to tune the PID for and the performance of system is compared

1. Trial and Error Method
2. Good Gain Method

1) Trial and Error Method

It is a trial and error method[5] but a computational stability rule is needed to set a mark for its effect. This is done by using the R-H stability rule.

R-H Criteria :

Keeping K_P part, set T_i and T_d to infinite and zero respectively, controller gain K_C could be obtained that would sustain the oscillation output which is called the ultimate gain, K_{CU}. For proper oscillations, K_C is set to be less than K_{CU}.

$$1 + K_{CU} * G(s) = 0$$

$$G(s) = \frac{32.0816}{5.1624 * 10^{-6} s^2 + 0.0605 s + 1}$$

$$1 + K_{CU} * \frac{32.0816}{5.1624 * 10^{-6} s^2 + 0.0605 s + 1} = 0$$

$$5.1624 * 10^{-6} s^2 + 0.0605 s + 1 + 32.0816 * K_{CU} = 0$$

Routh's array,

S^2	$5.1624 * 10^{-6}$	$1 + 32.0816 * K_{CU}$
S^1	0.0605	0
S^0	$1 + 32.0816 * K_{CU}$	

For no change in first column,

$$1 + 32.0816 * K_{CU} > 0$$

$$32.0816 * K_{CU} > -1$$

$$K_{CU} > - 0.03117$$

The above result shows that the main values of K_{CU} are greater than zero. With a trial and error tuning, the value of K_P can be set to num of the system transfer function i.e. 32.0816. The value of K_I is the inverse value of 0.03117 and the value of K_D is equal to the 0.03117.

So, from trial and error method the obtained value of PID parameters are K_P= 32.0816, K_I = 32.0821, K_D = 0.03117.

2) Good Gain Method

The Good Gain Method[14] is a simple method based on the experiments similarly to a trial and error method. It can be implemented on a simulation set up or on a real system. For tuning the parameters of PID controller, follow the steps given below.

1. Initially, set $T_i = \infty$ and $T_D = 0$ and keep increasing the value of K_P until satisfactorily stable response is obtained i.e. slight overshot and undershoot.

2. Find the value of T_{OU} i.e. the time difference between first overshoot and first undershoot of the response.

3. Calculate the value of T_i

$$T_i = 1.5 * T_{OU}$$

4. Calculate the value of T_D

$$T_D = \frac{T_i}{4}$$

For the given system, the values of the parameters of PID controller are K_P= 12, K_I = 1350.074, K_D = 1.851 * 10^{-4}

The Simulink model for the closed loop PID controller is shown in figure 2. Here, Trial and Error method and Good Gain methods have been implemented for the tuning of PID parameters.

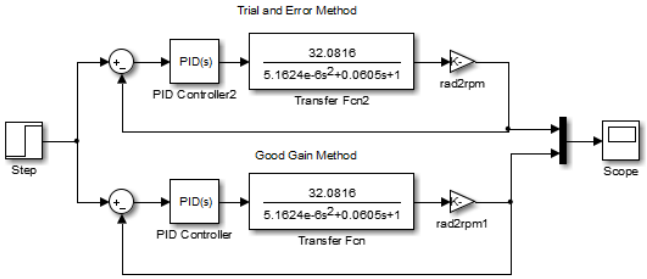

Fig. 2 : Simulink model for the closed loop PID controller

The comparison of the responses of both trial and error method and good gain method is shown in figure 3.

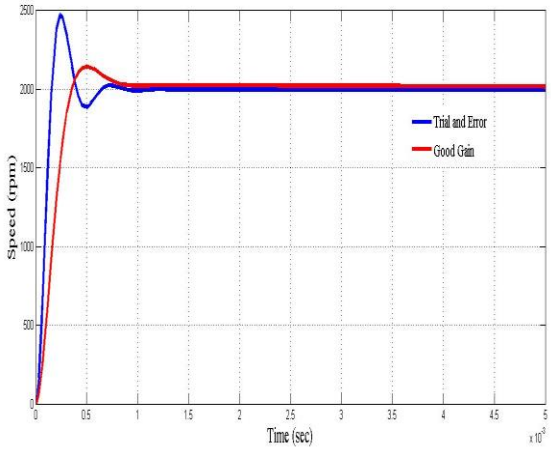

Fig. 3 : Responses of PID Controller

B. Fuzzy Logic Control

The concept of Fuzzy Logic was conceived by Prof. Lotfi A. Zadeh at the University of California at Berkley in 1965. Fuzzy logic is basically a multivalued logic that allows intermediate values to be defined between conventional evaluations like yes/no, true/false, etc. Notions like rather warm or pretty cold can be formulated mathematically and algorithmically processed. In this way an attempt is made to apply a more humanlike way of thinking in the programming of computers ("soft" computing).

FIS contains three components [4]:

1. Fuzzifier : The fuzzifier takes input values and determines the degree to which they belong to each of the fuzzy sets via membership functions of fuzzy logic system.
2. Rule base : The rule base contains linguistic rules that are provided by experts. It is also possible to extract rules from numeric data. Once the rules have been established, the FIS can be viewed as a system that maps an input vector to an output vector.
3. Defuzzifier : The Defuzzifier takes the values from fuzzy sets via membership functions and gives the crisp output.

Simulink model for the Fuzzy Logic controller is shown in figure 4.

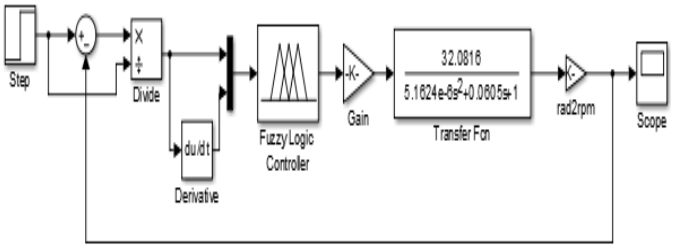

Fig. 4 : Simulink model for the Fuzzy Logic controller

The response of the Fuzzy Logic Control is shown in figure 5.

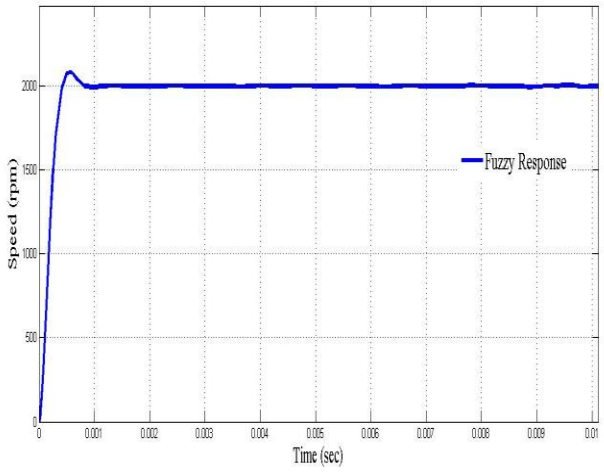

Fig. 5 : response of the Fuzzy Logic Control for speed 2000 rpm

C. Fuzzy + PID Control

Conventional PID controller does not give acceptable performance for systems with uncertain dynamics, time delays and non-linearity[11]. Hence it is necessary to automatically tune the PID parameters for obtaining satisfactory response. The automatic

tuning of PID controller has been done using fuzzy logic. Based on expert knowledge a fuzzy logic system transforms a linguistic control strategy into an automatic control strategy. Figure 6 shows the block diagram of a fuzzy PID controller. The fuzzy PID controller has been implemented using fuzzy logic toolbox in MATLAB.

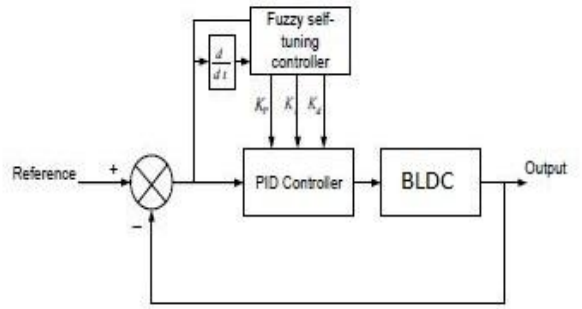

Fig. 6 : Block diagram of Fuzzy + PID Controller

Simulink model for the Fuzzy Logic controller is shown in figure 7.

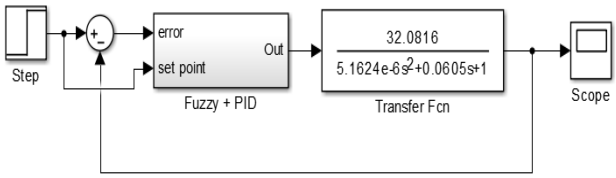

Fig. 7 : Simulink model for the Fuzzy + PID Logic

The response of the Fuzzy Logic Control is shown in figure 8

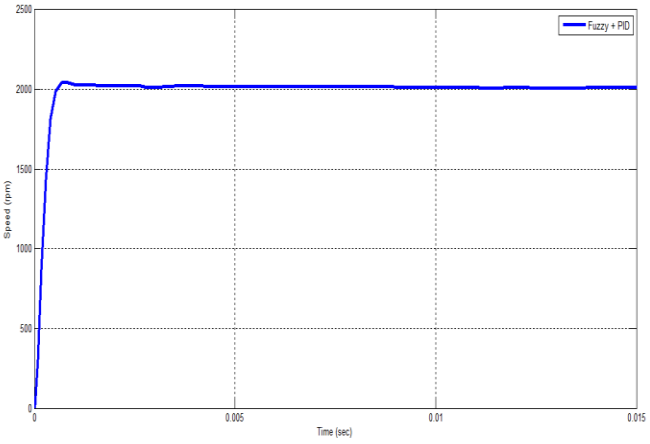

Fig. 8 : response of the Fuzzy + PID Logic for speed set at 2000 rpm

D. ANFIS Control

ANFIS is a multi-layer adaptive neural network-based fuzzy inference system[12]. ANFIS algorithm is composed of fuzzy logic and neural networks with 5 layers to implement different node functions to learn and tune parameters in a fuzzy inference system (FIS) structure using a hybrid learning mode. In the forward pass of learning, with fixed premise parameters, the least squared error estimate approach is employed to update the consequent parameters and to pass the errors to the backward pass. In the backward pass of learning, the consequent

parameters are fixed and the gradient descent method is applied to update the premise parameters. Premise and consequent parameters will be identified for membership function (MF) and FIS by repeating the forward and backward passes. Adaptive Neuro-Fuzzy Inference Systems are fuzzy Sugeno models put in the framework of adaptive systems to facilitate learning and adaptation [12]. Such framework makes FLC more systematic and less relying on expert knowledge. To present the ANFIS architecture, let us consider two-fuzzy rules based on a first order Sugeno model:

Rule 1: if (x is A1) and (y is B1) then (f1 = p1x + q1y + r1)
Rule 2: if (x is A2) and (y is B2) then (f2 = p2x + q2y + r2)

where x and y are the inputs, Ai and Bi are the fuzzy sets, fi are the outputs within the fuzzy region specified by the fuzzy rule, pi, qi and ri are the design parameters that are determined during the training process.

Out of the five layers, the first and the fourth layers consist of adaptive nodes while the second, third and fifth layers consist of fixed nodes. The adaptive nodes are associated with their respective parameters, get duly updated with each subsequent iteration while the fixed nodes are devoid of any parameters. The ANFIS architecture to implement these two rules is shown in Fig. 9.

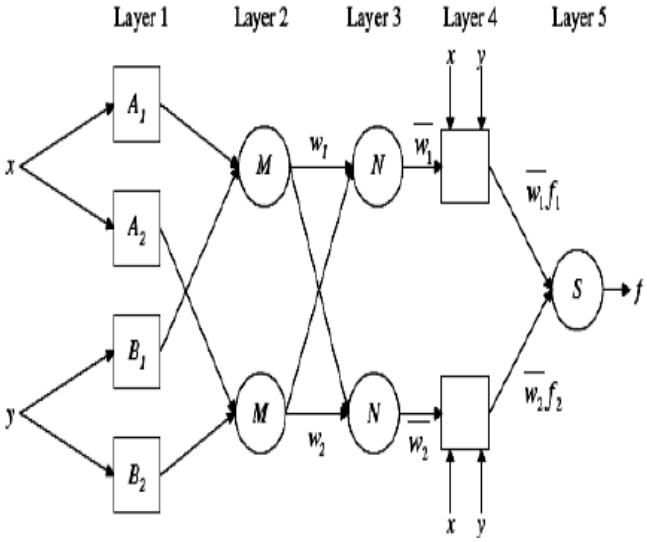

Fig. 9 : ANFIS architecture

Layer 1: fuzzification layer Every node I in the layer 1 is an adaptive node. The outputs of layer 1 are the fuzzy membership grade of the inputs, which are given by:

$$O_{i1} = \mu A_i(x), \text{ For } i=1,2 \quad (3)$$

$$O_{i1} = \mu B_{i-2}(y), \text{ For } i=3,4 \quad (4)$$

where x and y is the inputs to node i, where A is a linguistic label (small, large) and where μAi (x), μBi-2 (y) can adopt any fuzzy membership function.

Layer 2: rule layer a fixed node labeled M whose output is the product of all the incoming signals, The outputs of this layer can be represented as:

$$O_{i2} = w_i = \mu A_i(x)\,\mu B_i(y) \quad i=1,2 \quad (5)$$

Layer 3: normalization layer are also fixed node is a circle node labeled N.

$$O_{i3} = \_w_i = w_i / (w1 + w2) \quad i=1,2 \quad (6)$$

Layer 4: defuzzification layer an adaptive node with a node The output of each node in this layer is simply the product of the normalized firing strength and a first order polynomial.

$$O_{i4} = \_w_i f_i = w_i (p_i x + q_i y + r_i) \quad i=1,2 \quad (7)$$

Layer5: summation neuron a fixed node which computes the overall output as the summation of all incoming signals.

$$O_{i5} = \Sigma \_w_i f_i = \Sigma_{i=1} w_i f_i / (w1 + w2) \quad (8)$$

Simulink model for the ANFIS controller is shown in fig 10.

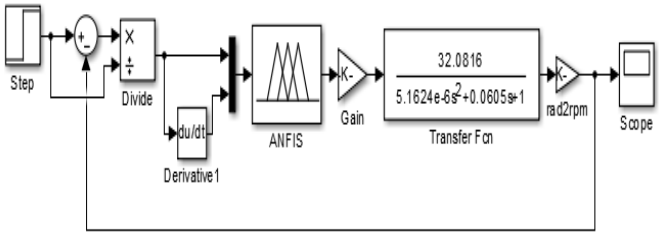

Fig. 10 : Simulink model for the ANFIS controller

The response of the ANFIS Controller is shown in fig 11.

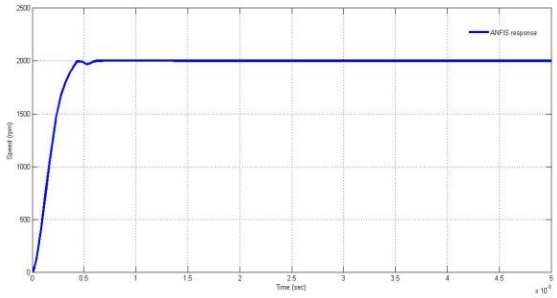

Fig. 5 : response of the Fuzzy Logic Control for speed 2000 rpm

IV. RESULT

The transient parameter of the system for the different tuning parameters and Intelligent Controllers are obtained and shown in Table 1.

Table 1 : Transient Response for Different Control Schemes

Control Schemes	Transient Parameters		
	Rise Time (ms)	Settling Time (ms)	Peak Overshoot (%)
Trial and error	0.1253	0.7632	23.5
Good Gain	0.24	3.745	7.1
FLC	0.339	0.6949	4.05
Fuzzy + PID	0.4101	0.6449	2.05
ANFIS	0.2764	0.4307	-

From Table 1, it is clearly visible that the settling time and peak overshoot of the system for the ANFIS Control is much better than that of other control schemes.

V. CONCLUSION

With the help of simulation results and the data presented, it can be concluded that response of the system is better when PID controller parameter is tuned by GGM instead of trial and error method. The result also reveals that the application of soft computing techniques like Fuzzy, Fuzzy+PID and ANFIS to the system can give a better performance than the conventional controller.

REFERENCES

[1] Abhishek D. Gandhi, Suryaprakash Singh, Bhautik M. Daxini, "Speed Control of Brushless DC Motor Using PID and Fuzzy Controller", ETCEE, IJAERD, March-2015.
[2] C. SheebaJoice, P. Nivedhitha, "Simulation of Speed Control of Brushless DC Motor with Fuzzy Logic Controller", International Journal of Electrical, Electronics and Data Communication, Volume-2, Issue-4, April-2014.
[3] Pooja Agarwal, Arpita Bose, "Brushless Dc Motor Speed Control Using Proportional- Integral and Fuzzy Controller", IOSR-JEEE, Volume 5, Issue 5 (May. - Jun. 2013), PP 68-78.
[4] Umesh Kumar Bansal and Rakesh Narvey, "Speed Control of DC Motor Using Fuzzy PID Controller", Advance in Electronic and Electric Engineering, Volume 3, Number 9 (2013).
[5] Oludayo John Oguntoyinbo, "PID Control of Brushless DC Mptpr and Robot Trajectory Planning and Simulation with MATLAB/SIMULINK," Degree Program of IT, Vaasa University of Applied Sciences, Finland, December 2009.
[6] Laurent Foulloy, Sylvie Galichet, "Fuzzy Control with Fuzzy Inputs", IEEE TRANSACTIONS ON FUZZY SYSTEMS, VOL. 11, NO. 4, AUGUST 2003.
[7] P.C. Sen, "Principles of Electric Machines and Power Electronics", John Wiley & Sons, 1997.
[8] Karl J. Astrom and Tore Hagglund, "PID Controllers : theory, design and tuning", second edition.
[9] Takashi Kenjo, Shigenobu Nagamori, "Permanent-magnet and brushless DC motors", volume 18, Oxford University Press,16 Jan 1985.
[10] Padmaraja Yedamale, ―Brushless DC (BLDC) Motor Fundamentals‖, Microchip Technology Inc. 2003.
[11] Vikram Chopra, Sunil K. Singla, Lillie Dewan, "Comparative Analysis of Tuning a PID Controller using Intellegent Methods", Acta Polytechnica Hungarica, Volume 11, No.8, 2014.
[12] J.R. Jang, "ANFIS: Adaptive-network-Based Fuzzy Inference System",IEEE Trans. On Systems, Man and Cybernetics, Vol. 23, No.3, May.1993, pp.665-685.
[13] Fuzzy Logic Toolbox User's Guide © copyright 1995-1999 by The MathWorks, Inc.
[14] Finn Haugen, "The Good Gain Method for PI(D) Controller Tuning", 19 July, 2010.
[15] George W. Younkin, "Electric Servo Motor Equations and Time Constants".
[16] http://www.maxonmotor.in/maxon/view/product/motor/ecmotor/ecflat/ecflat45/200141.

