Proceedings of Engineering and Technology Innovation, vol. 31, 2025, pp. 58-72

Performance Trade-Offs in AI-Based Speed Control of PMDC Motors: A Comparative Study of Fuzzy Logic and Neural Network Controllers

Hayder Salim Hameed*

Department of Electrical Power and Machines Engineering, College of Engineering, University of Diyala, Baqubah, Iraq

Received 09 April 2025; received in revised form 05 July 2025; accepted 10 July 2025

DOI: https://doi.org/10.46604/peti.2025.15024

Abstract

This paper aims to compare the speed control of DC motors using two distinct artificial intelligence controllers: fuzzy logic controllers (FLC) and artificial feedforward neural networks (AFFNN). This study presents the first comprehensive comparison of FLC and ANN under identical test conditions, offering actionable guidelines for industrial applications. The driving system has been designed and tested using MATLAB/Simulink. Simulation results show that the AFFNN controller's rise time at 130 (rad/s) is 14.9 ms, whereas the fuzzy logic controller's is 32.4 ms. Furthermore, the neural network controller and fuzzy logic controller overshoot by 2.6e-06% and 0%, respectively. However, the neural controller takes 213.5 ms to reach its peak, whereas the fuzzy controller achieves this level earlier, at 94.6 ms. AFFNN gives a faster rise time and minimal settling time. However, FLC gives a faster peak response with zero overshoot for effective PMDC motor control.

Keywords: permanent magnet DC motor, speed control, fuzzy logic controller, artificial neural network

1. Introduction

Permanent Magnet DC (PMDC) motors are very important in many industrial settings because they are very reliable, efficient, and easy to handle. They are used in robotics applications like surgical robots, where having precise motion control is vital for ensuring patient safety. In the world of electric vehicles (EVs), efficient motor control plays a crucial role in determining both the range and performance of the vehicle. This is also true for resource-limited embedded systems, where computational efficiency is key. Take modern EVs like those from Tesla and Rivian, for example; their PMDC motor control systems need to strike a balance between performance and energy efficiency to ensure maximum driving range while still providing quick acceleration. The same principle applies in industrial automation, where PMDC motors in CNC machines demand nanometer-level precision control to effectively manufacture semiconductor components.

One of the most critical characteristics that a DC motor must regulate is rotation speed. PID control techniques are often employed using a variety of approaches to regulate DC motor speed. PID (proportional-integral-derivative) is a common control approach used in control engineering. PID control is a combination of three kinds of controllers: proportional, integral, and derivative. Parameters may influence the quality of a control response. PID control uses a feedback mechanism to rectify inaccuracies in the measurement error value and the deviation value. In general, the PID control system may be employed simultaneously or individually since each control has its benefits, such as speeding rise time, decreasing steady-state errors, and lowering overshoot or undershoot. Technological advancements are starting to move toward an automated approach that uses computers as the control center [1].

All the foregoing is constrained by advanced control mechanisms such as fuzzy logic controllers (FLCs) and artificial

* Corresponding author. E-mail address: haydersalim@uodiyala.edu.iq

neural networks (ANNs). FLCs handle nonlinearities and instabilities in the system very efficiently to simulate human thinking and apply language norms. Research has shown that the fuzzy logic controller is effective in improving the performance of the DC motor speed control system, such as with more robust disturbance rejection and reduced steady-state errors. The FLC's incorporation in a DC motor speed control system gave rise to better adaptability with less sensitivity to variations in design parameters [2].

In the same manner, artificial neural networks have been used in controlling the speed of a DC motor due to their ability to learn and model highly nonlinear relationships, as they were inspired by the neural architecture of the human brain. An artificial neural network-based controller can adapt to changing system dynamics since it does not require explicit mathematical models, thus making it applicable to cases wherein system parameters are uncertain or vary with time. Studies have shown that ANN-based controllers improve both transient and steady-state performances of a DC motor while providing features like reduced overshoot and better response time [3].

Both FLCs and ANNs have put forward a potential enhancement over classic PID controllers. Each opposing method has its inherent advantages and challenges. FLCs provide greater interpretability and ease of implementation but require substantial expertise for setting up proper rules and membership functions. On the other hand, artificial neural networks focus on flexibility and learning but require excessive computational resources and a large amount of training data to perform just about adequately [4]. Speed control precision becomes a key factor in PMDC motors for industrial applications. Thus, it becomes imperative to test and compare such advanced control techniques to conclude their use under all operating conditions.

2. Literature Review

From the researchers' point of view, speed controllers for DC motors have been a topic of strong research for many years, as they serve a versatile utility in various industrial processes. The proportional-integral-derivative (PID) controller is still preferred; however, studies reveal that it shows certain problems, such as overshoot and others, which can include a slower response to a change in torque requirements and sensitivity to setting changes, as discussed in [5].

Therefore, alternative solutions have been proposed, such as fuzzy logic and fractional-order PID (FOPID) algorithms. In [6], a better fractional-order $PI^{\lambda}D^{\mu}$ (FOPID) controller was discussed. It was multiplexed for the speed errors in permanent magnet DC motors (PMDC), which serve as wire feeder systems (WFSs) used in gas metal arc welding (GMAW) processes. This method eliminates speed variations and also overshoots, undershoots, and steady-state changes by implementing ant colony optimization-inducing controller tuning. Similarly, a fuzzy-logic-based PI control scheme is useful for automatically tuning PI parameters every time the command speed changes by one step, which leads to a better speed response.

Alset et al. [7] show an example where they used a fuzzy logic controller on a low-cost ATMEGA-328P-PU microcontroller. They pointed out its benefits compared to traditional PI solutions. Soliman et al. [8] give a different view on hardware designs by talking about a PMDC prototype built on a flexible microarchitecture that works well for IoT edge computing and has FPGA-based dynamic pipeline stages. In [9], an interior point algorithm (IPA) was used to figure out parameters in geared PMDC motors by looking at how they responded to changes in speed and current. This exercise turned the problem into an optimization task to make the parameters more accurate.

In [10], a PI+DF (Proportional Integral Derivative Filter) controller is tuned with a stochastic fractal search algorithm. A comparison shows that the SFS-tuned controller improves speed response and robustness in simulation and experimental tests on a DC servo system over PSO and the existing method. Reference [11] talks about the problems that can come up when picking the PWM frequency and pulse width. It also suggests a useful control function that makes sure the wire-feeding speed is correct in GMAW processes that use back-EMF feedback. In [12], an integral sliding mode convex optimization approach is analyzed. This plan uses a sub-gradient method that averages over time to find the best loss function for uncertain LaGrange systems powered by PMDC motors in real-time. Sliding mode control, another robust control method, appears in [13], where the authors incorporate a washout filter (SMC-w) to regulate PMDC motor speed. They evaluate how the controller responds to variations in reference speed and load torque.

Fuzzy logic and hybrid fuzzy-PID control methods are also looked at. For example, in [14], PID controllers are compared to fuzzy logic controllers (FLCs) for controlling the speed of a DC motor, looking at things like overshoot, rise time, and torque disturbances. The studies emphasize that fuzzy-based control techniques can surpass traditional PID controllers in certain scenarios, yet each method has unique trade-offs in complexity and tuning. Çelik and Öztürk [15] present a new fuzzy logic estimator (FLE) to reduce current pulsation during the operation of a high-speed brushless DC motor. The proposed approach uses the commutation angle to ensure equal current slew rates, thus eliminating the need for a torque observer or detection. The FLE rule base was optimized using a genetic algorithm. The results of simulations show that there is a reduction in current pulsation and an improvement in the output power of the motor.

In [16], the authors compare the five different control strategies, such as conventional (Root Locus, Ziegler-Nichols) and intelligent (Fuzzy Logic, Genetic Algorithm, and online PI tuning) methods in speed and position control of DC motors. The performance of each controller is compared to transient response, disturbance rejection, and robustness through simulation. The comparison was then done, using standard indices, to bring out the relative merits and demerits of each of the approaches. In [17], researchers create two versions of an ANN-based PID controller (ANNPID) to control the speed and position of a PMDC motor. Simulations show that they work better than traditional PID controllers in terms of transient response and robustness. Another advanced approach is the use of nonlinear extended state observers.

In [18], the active disturbance rejection control (ADRC) framework gets a new finite-time NLESO to deal with unknowns and disturbances that come and go. This improvement makes peaking effects much less common in linear ESOs. Fuzzy-PID hybridization is used again in [19], where the authors compare three different ways to control the speed of a PMDC motor when it is under load. This study shows how fuzzy logic can automatically tune a PID controller. Adaptive neuro-fuzzy inference systems (ANFIS) can be useful. In [20], neural networks and fuzzy logic combine to control the speed of a DC motor. An Arduino Due board is used for real-time implementation, and the hybrid system performs well under parameter uncertainties. The authors applied the Levenberg-Marquardt backpropagation technique in Simulink/MATLAB and tested the FNN on an unidentified motor at various reference speeds. Despite excluding some real-world elements from the simulations, the authors noted both advantages and potential disadvantages of using FNN-based control for PMDC motors.

At last, Çelik et al. [21] show a nonlinear method of PI control whereby an exponential PI (EXP-PI) controller is built. The Sparrow Search Algorithm (SSA) for offline parameter optimization is chosen, as its exponential feature improves the speed control of PMDC motors. Table 1 includes a summary of prior methods and their limitations.

Table 1 Summary of prior methods and their limitations

Ref.	**Prior methods**	**Limitations**
[5]	Conventional PID	Overshoot, delayed response, sensitive to settings
[7]	Fuzzy PI	Limited to small-scale embedded systems
[15]	FLC vs. PID	Needs careful rule base construction
[19]	ANN-PID	No real-time verification
[21]	Fuzzy-PID Hybrid	Still present parameter sensitivity

In this work, a hybrid learning approach using fuzzy logic was suggested to train a feedforward neural network (FFNN), combining the power of fuzzy logic control (FLC) and the adaptability of artificial neural networks (ANN). The FFNN replaces the fuzzy controller following offline training, significantly enhancing real-time execution speed. The system implementation and testing are conducted in MATLAB-Simulink, following embedded system realizability considerations in the design. Performance is rigorously examined in the context of regulation time, rise time, and overshoot for various load disturbances, establishing the method's robustness and practicability.

This research presents a ten-hidden-layer feedforward neural network (FFNN) controller of a fuzzy logic (FL) trained on artificial intelligence (AI). Generally, fuzzy logic (FL) is adjusted to enhance the performance and control of the PMDC motor speed under load and external disturbances. FFNN can learn its behavior utilizing offline learning between the inputs and outputs of the controller. After the learning was done, the neural network controller was switched out for the fuzzy controller to get a controller that could keep the system stable and make up for the problems that the fuzzy controller caused with the hardware implementation. The following tasks will be resolved by this work: MATLAB-Simulink is used to create mathematical models of the selected controller.

3. Mathematical Model of Permanent Magnet DC (PMDC) Motor

The conventional equivalent circuit of a DC motor includes a winding resistance of the armature Ra, a self-inductance of armature winding La, and an induced or back electromotive force (emf) e, as shown in Fig. 1 [22]. Eq. (1) defines the electrical equation of the time-domain behavior of a DC motor.

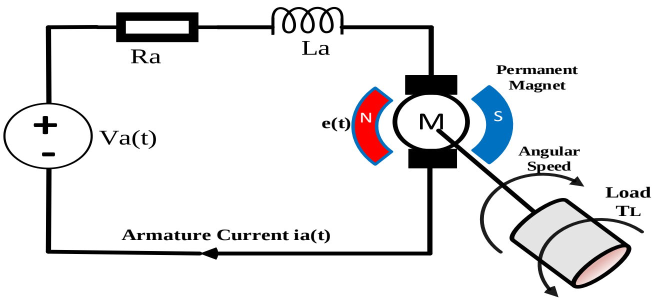


Fig. 1 Equivalent circuit of PMDC motor

$$V_a(t) = R_a . i_a(t) + L_a \frac{d}{dt} i_a(t) + e_b(t) \tag{1}$$

In this case, the armature voltage, induced back emf, armature winding inductance, armature resistance (R_a), and armature current (i_a) are represented by E and V, respectively. The induced emf equation is represented as:

$$e_b(t) = K_e \omega(t) \tag{2}$$

The electromagnetic torque of the motor is denoted as T_m, while the torque due to external load is represented by T_L, which are provided by

$$T_m(t) = J \frac{d\omega(t)}{dt} + B\omega(t) + T_L(t) \tag{3}$$

$$T_m(t) = K_t i_a(t) \tag{4}$$

Here, J stands for the motor's inertial moment with viscous friction coefficient B, and K_t denotes the torque constant (typically $K_t = K_e$).

The following equations are described in state-space form, where $i_a(t)$ and $\omega(t)$ are the state variables:

$$\frac{di_a(t)}{dt} = \frac{1}{L_a}(V_a(t) - R_a i_a(t) - K_e \omega(t)) \tag{5}$$

$$\frac{d\omega(t)}{dt} = \frac{1}{J}(K_t i_a(t) - B\omega(t) - T_L(t)) \tag{6}$$

Taking Laplace transform with respect to Eq. (1) and Eq. (3), it yields:

$$v_a(s) = R_a.i_a(s) + sL_a I_a(s) + K_e \omega(s) \tag{7}$$

$$sJ\omega(s) + B\omega(s) = K_t I_a(s) \tag{8}$$

After some calculations, the open-loop transfer function is taken, which takes the rotor speed as output and the voltage at its ports (V_t) as input [23]:

$$\frac{\omega(s)}{V_t(s)} = \frac{K_t}{L_a J s^2 + (R_a J + L_a B)s + R_a B + K_t K_e} \tag{9}$$

4. Fuzzy Logic

A subfield of artificial intelligence known as fuzzy logic uses a collection of loose rules based on continuously fluctuating input data to control the output signal within predetermined bounds. Unlike linear laws, which only work in a specific [24] operating mode, fuzzy logic laws can function and guarantee the desired quality of control operations in all engine operating modes, including direct current. A fuzzy controller works by using a set of rules that represent the connection between the input data and the output signal. Several components build the FLC, aiding in the design process. Fig. 2 depicts the architecture of the fuzzy control system [25].

The Fuzzy Logic Controller (FLC) operates through four key components: the rule base, fuzzification, inference, and defuzzification. The rule base consists of a predefined set of rules that govern how the system should be controlled based on expert knowledge. Control actions would be derived from applying logical rules to these fuzzy inputs. Defuzzification entails converting these fuzzy outputs into numerical values to use them for controlling the system or plant. These phases present an intuitive and flexible control approach suitable for nonlinear and complex systems.

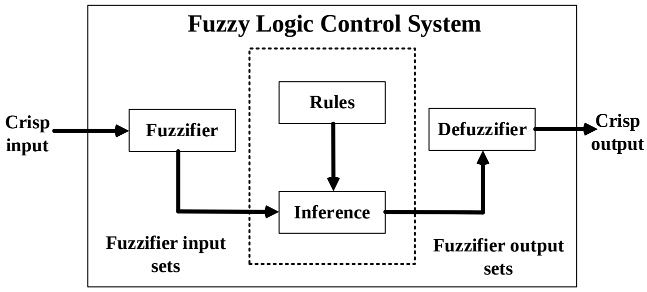
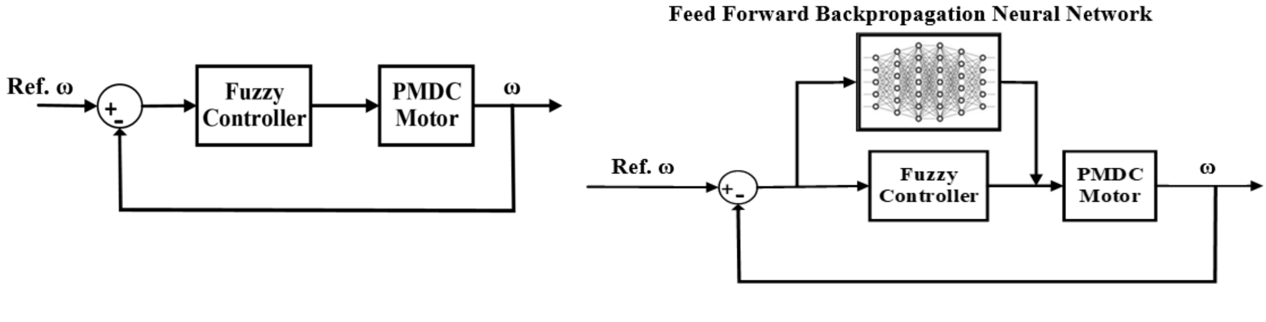


Fig. 2 Fuzzy Control System

5. The Development of Artificial Networks

Artificial neural networks (ANNs) are one of the newest technological systems that efficiently learn patterns from training data. ANNs were trained to simulate a function, demonstrating the input and output functions to reduce the error between the expected and actual outputs. The training for the backpropagation approach computes the patterns of output and modifies the weights. The ANN controller was designed by first performing and verifying a fuzzy logic controller, as depicted in Fig. 3(a). Then, the ANN controller was modified to effectively replace the original FL controller by using the input/output information from the performed FLC as the input and target data for the learning algorithm, as depicted in Fig. 3(b) [26-27].

The speed of the PMDC motor serves as the ANN's only input layer. One hidden layer and ten more hidden layers constitute the hidden layer development. Nonlinear input and output transformations were carried out using the sigmoid activation function. Tansig is the activation function of the hidden layer, while purelin is the activation function of the output. The purpose of the ANN's weight is to reduce the discrepancy between the neural network's intended and actual outputs. The acquired input data set is used to train the neural network. The procedure of training a neural network is known as functional fitting, using a collection of inputs, such as the speed of the PMDC motor, to create a matching set of target outputs, such as the armature voltage of the PMDC motor. Following the data fit, the neural network produces a generalization of the input-output connection [28].



(a) Fuzzy controller performing (b) Replacing the original fuzzy controller

Fig. 3 Design of ANN controller

6. MATLAB Simulink Implementation

The implementation of fuzzy logic and neural network controllers for a permanent magnet DC (PMDC) motor in MATLAB Simulink requires creating a comprehensive simulation system that includes motor simulation, intelligent control methods, and performance evaluation. The major goal is to build, simulate, and evaluate such controllers under a variety of operational situations, including abrupt speed changes and disturbances to load. The Simulink implementation is broken out in great depth here, including motor modeling, controller design, and assessment criteria.

6.1. PMDC Motor Simulation

Using basic Simulink blocks like integrators, gain blocks, and sum blocks, the electrical and mechanical equations are constructed as part of the Simulink implementation. A specialized PMDC motor subsystem was created to provide flexibility and modularity, allowing for smooth integration with different controllers, as depicted in Fig. 4. The operational and physical properties of the PMDC motor for simulation testing are shown in Table 2.

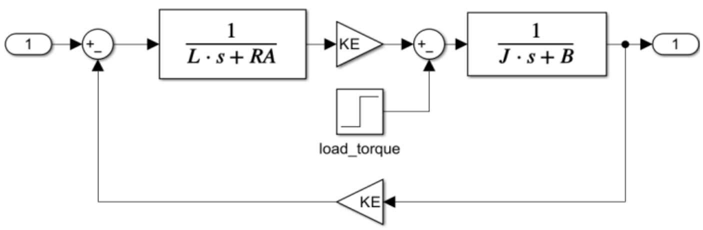
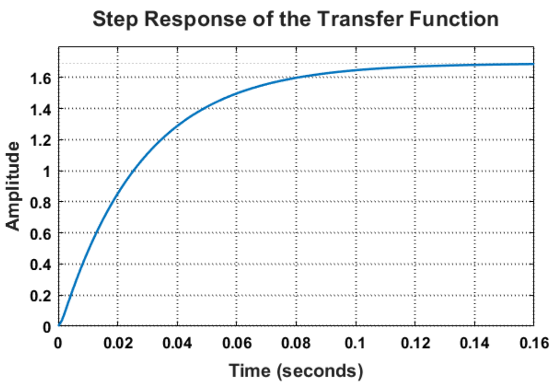


Fig. 4 Simulink block diagram of the PMDC motor model

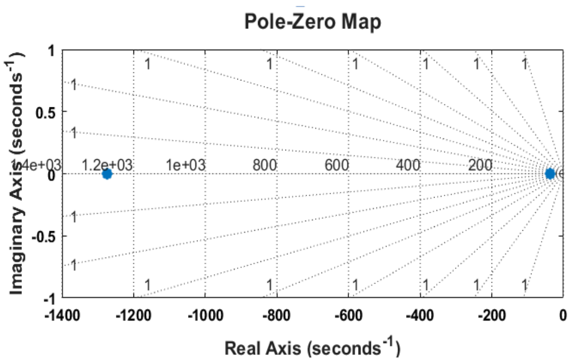
Table 2 The operational and physical properties of the PMDC

Quantity	Value	Unit
Armature Resistance Ra	4.98	Ω
Armature Inductance La	3.8	mH
Inertia Moment J	0.0019258	$Kg.m^2$
Friction Coefficient B	0.00171	N.m/(rad/s)
Torque Constant K_T	0.5766	N.m/A
Back Electromotive Force Constant Ke	0.577	V/(rad/s)
Rated voltage	12	V
Rated Speed	201	rad/s

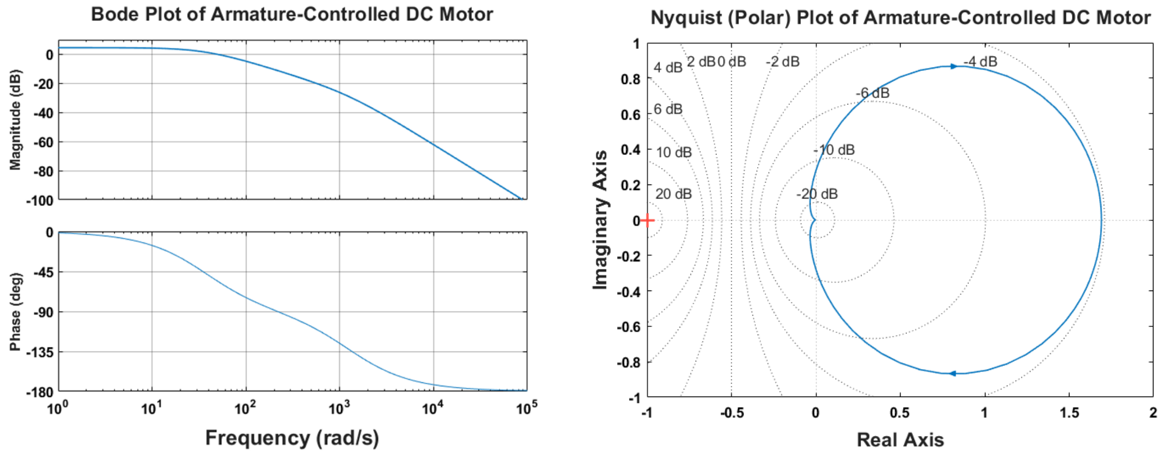
Based on the PMDC motor transfer function, the step response is plotted. To assess system stability, pole-zero plots and root locus plots are done by means of MATLAB. In both cases, the phase margin and the gain margin are infinite, i.e., the system is robust and has an ignorable overshoot. As shown in the polar plot in Fig. 5, the poles lie on the left-handed side of the s plane. The system is therefore stable.



(a) Step response of controlled PMDC motor



(b) Pole-zero plot of the controlled PMDC motor



(c) Bode plot and Polar plot

Fig. 5 A formal stability analysis PMDC motor

6.2. Constructing a Fuzzy Logic Controller (*FLC*)

FLC is developed and simulated in MATLAB/Simulink using the Sugeno model with CoG (Center of Gravity) for defuzzification. In Simulink, the Fuzzy Logic Toolbox is used, where the FIS Editor sets up fuzzy groups, rules, and reasoning methods, and the Fuzzy Controller block works with the PMDC motor system for testing and performance analysis, as depicted in Fig. 6. Table 3 displays the rules that connect the inputs to the fuzzy output, where PL = Positive Large, PS = Positive Small, ZE = Zero, NS = Negative, and NL = Negative Large Small.

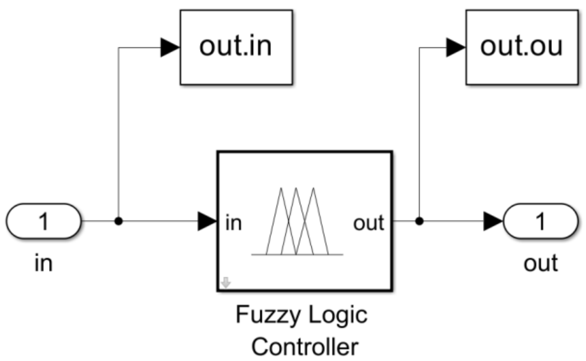
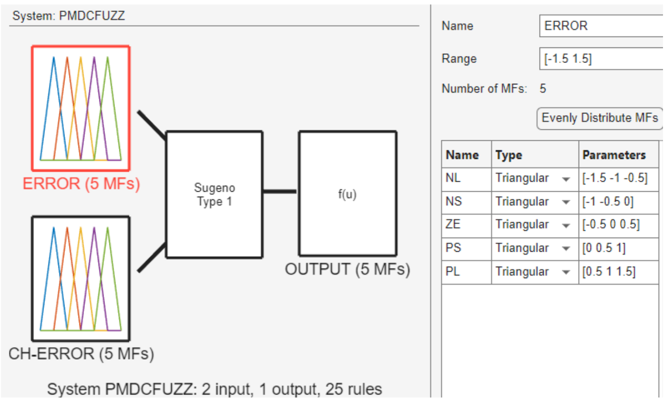


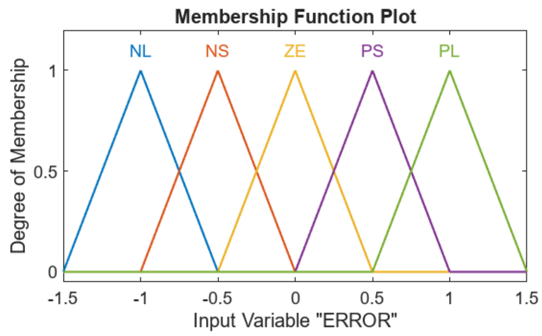
Fig. 6 Obtaining the dataset with Simulink

Table 3 Fuzzy Rule Matrix

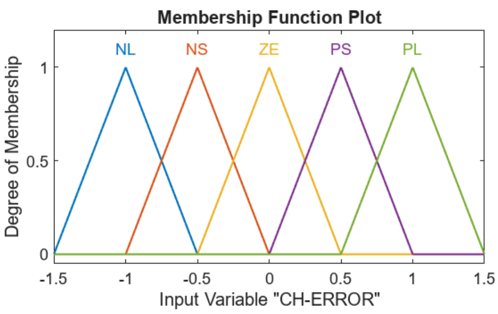
CH-ERROR \ ERROR	NL	NS	ZE	PS	PL
NL	NL	NL	NS	NS	ZE
NS	NL	NS	NS	ZE	PS
ZE	NS	NS	ZE	PS	PS
PS	NS	ZE	PS	PS	PL
PL	ZE	PS	PS	PL	PL



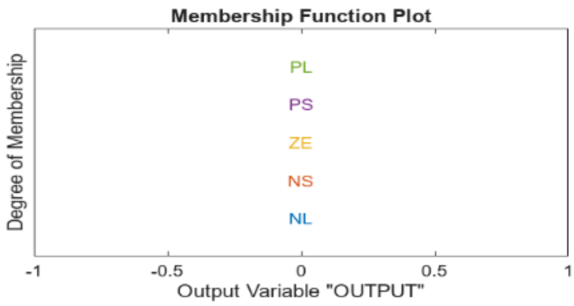
(a) Membership functions and variable settings



(b) ERROR

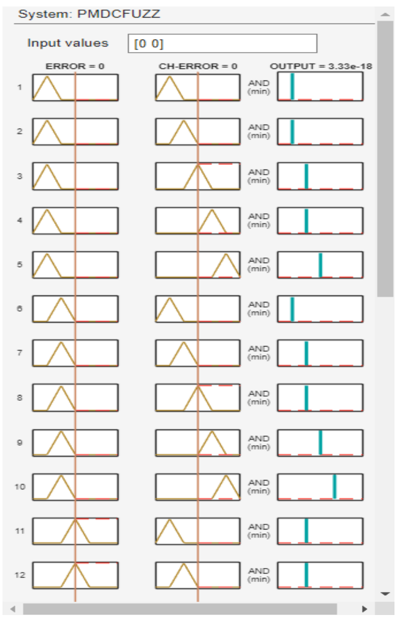


(c) CH-ERROR

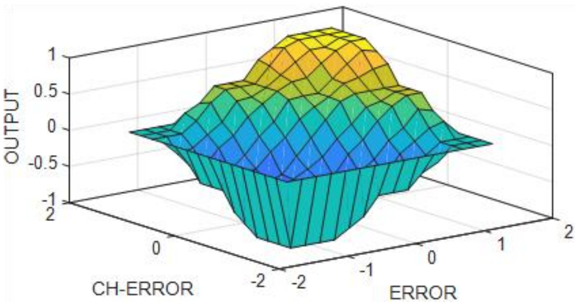


(d) OUTPUT

Fig. 7 Variables for Fuzzy Input "ERROR"



(a) Rule Viewer



(b) Fuzzy rule surface

Fig. 8 Fuzzy rule viewer and surface

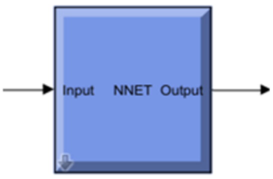
During simulation, the range of fuzzy membership function parameters is identified by trial and error. There are twenty-five rules in the fuzzy function's approximation law. The Max-Min approach, which Sugeno devised, is the inference technique employed. This approach is founded on a comprehensive comprehension of the system's behavior. Fig. 8(a) looks like a visual representation of how a Fuzzy Logic Controller (FLC) works for the PMDC motor system, showing how it uses fuzzy rules to process the input values of Error and Change in Error (CH-ERROR) to decide the control output. From Fig. 8(b), it is clear that the use of twenty-five rules is optimal for a given engine since the mismatch error between the control and desired signals is minimal. An additional rise in the number of rules does not influence the extent of the mismatch; however, it does lead to an increase in regulation time and a broader range of both positive and negative errors.

The fuzzy controller has two inputs and one output. The output is the speed, whereas the input variables are error (ERROR) and derivative error (CH-ERROR). A triangle membership function has been decided on for the fuzzification process due to its simple computation. The fuzzy input and output membership functions and variable settings are shown in Fig. 7.

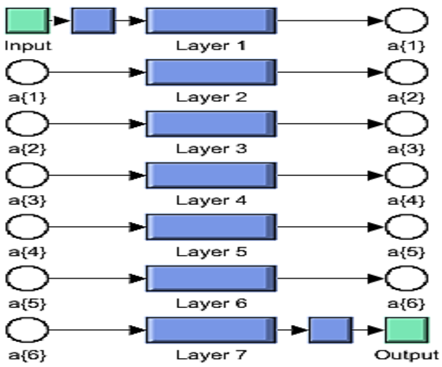
6.3. Neural Network Controller (ANN) Implementation

The Artificial Neural Network (ANN) Controller is designed to predict the optimal control voltage required to regulate motor speed efficiently, learning from historical data rather than relying on predefined rules like a fuzzy logic controller (FLC). The training dataset for the neural network controller was created utilizing MATLAB/Simulink simulations of a PMDC motor with a fuzzy controller, as depicted in Fig. 6. The ANN received the speed error e(t) and the change in error Δe(t) as inputs and the control voltage u(t) as output. Simulations consisted of step reference inputs of 0 to 130 rad/s and load disturbances. About 12,501 samples were collected at a sampling interval of 2.5 ms. Data split into training, validation, and testing sets guarantees correctness by using the Levenberg-Marquardt backpropagation method. The dataset was normalized and split into three subsets: 80% training, 10% validation, and 10% testing. The data set was chosen since it covers dynamic motor activity under normal operating settings, supporting generalizability in the trained ANN.

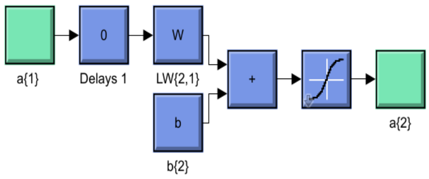
The neural network design includes an input layer that absorbs speed error, change in speed error, and previous control values, followed by one or more hidden layers with activation functions like ReLU or sigmoid, and an output layer that projects the control voltage. Once trained, the ANN is used in Simulink by combining the neural network block, enabling the controller to evaluate speed error inputs and forecast a suitable voltage for motor control. The neural network function-fitting model in Simulink is illustrated in Fig. 9.



(a) A custom neural network



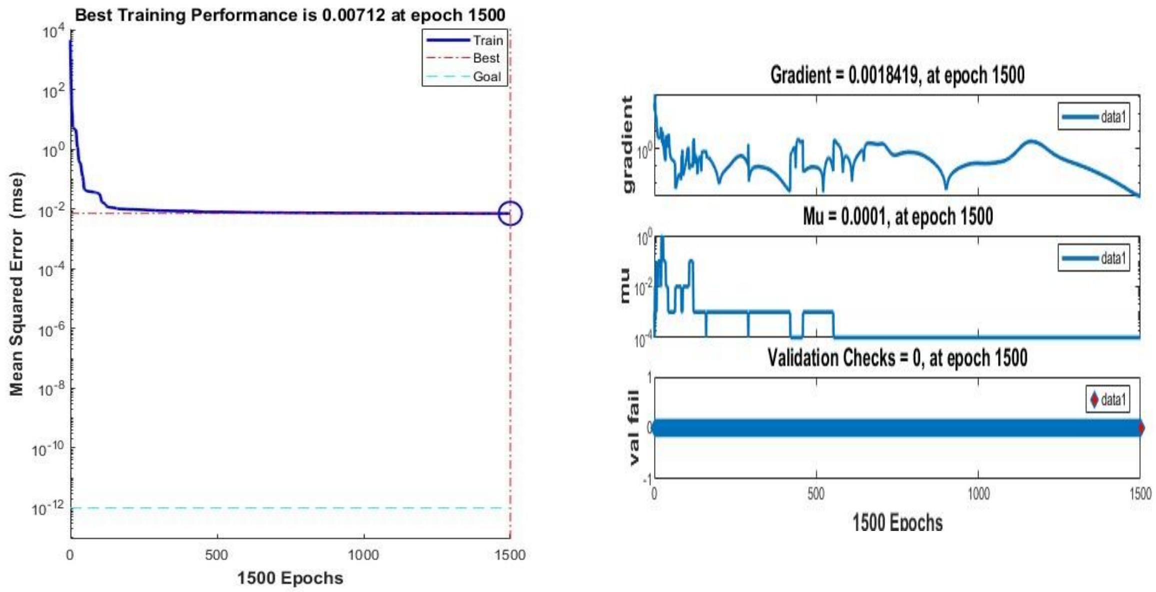
(b) Subsystem of Custom Neural Network



(c) Layer 2 subsystem

Fig. 9 ANN Controller's Hidden Number

For this project, the Levenberg-Marquardt backpropagation technique and ten hidden layers were used to train the ANN controller to function well. The ANN model was trained with a learning rate set at 0.06, and the hyperparameters were fine-tuned by tweaking the number of epochs, activation functions, and weight initialization. As seen in Fig. 10(a), the model showed a steady learning pattern, with the mean squared error (MSE) gradually decreasing and leveling off at 0.00712 by the 1500th epoch. This suggests that the training was effective and didn't lead to overfitting. Fig. 10(b) provides a look at the convergence metrics, where the gradient hit 0.0018419, and the MU term indicated stable learning dynamics.



(a) ANN Controller's Performance (b) ANN Controller's Training State

Fig. 10 ANN Training State

7. Simulation and Performance Comparison

To evaluate the efficacy of both controllers, Simulink runs several test scenarios, such as a step response test and a load disturbance test. The step response test gives the motor a step input for a speed change from 130 to 80 rad/s and checks overshoot, rising time (the time it takes to reach the speed), and settling time (the time it takes to stay stable within a small error margin). During the load disturbance test, the speed stability of the controllers is evaluated, and a sudden change in load torque is applied to evaluate the resilience of the system. To assess the robustness of the controllers, the noise rejection test models sensor errors and external disturbances using random noise. Fig. 11 illustrates the two-controller simulated circuit for a permanent magnet (PMDC) motor.

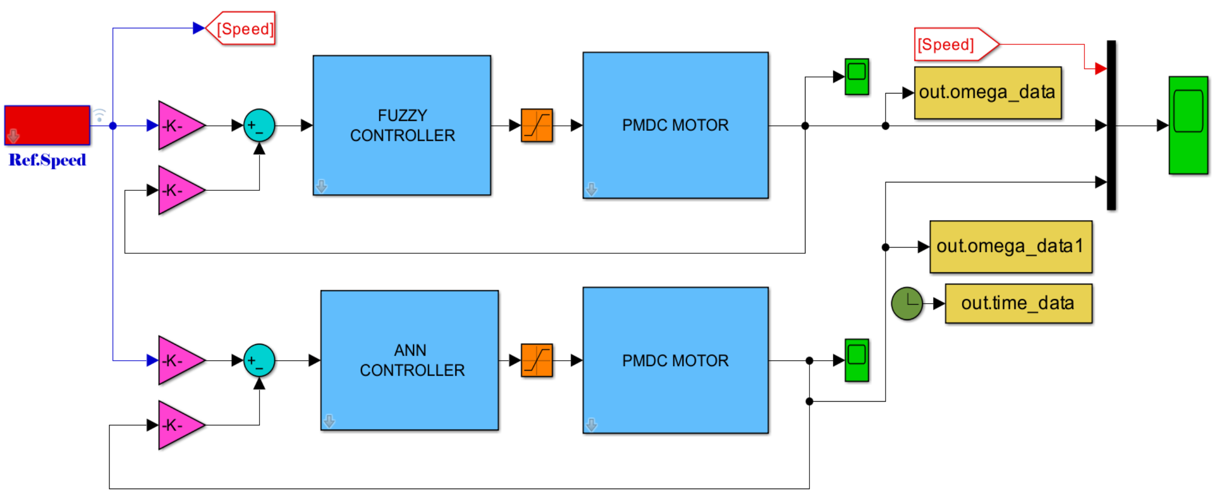


Fig. 11 MATLAB Simulink model of PMDC motor with FLC and ANN controllers

8. Results and Discussion

In this section, a comparison of the neural network controller (NNC) and fuzzy logic controller (FLC) is made in the speed control of a permanent magnet DC (PMDC) motor. Performance evaluation rests on key criteria such as speed response, resistance to load changes, disturbance rejection, and computational complexity. The behavior of the controllers under different conditions has been assessed through simulations on the MATLAB Simulink platform. Some significant parameters for evaluating the performance of fuzzy and neural network controllers include rise time, transient response, settling time, overshoot, steady-state error, and peak value. Both controllers were tested at two different speeds, 130 rad/s and 80 rad/s, for an in-depth study. Fig. 12 and Fig. 13 show the speed response for the two controllers at 130 rad/s and 80 rad/s, respectively.

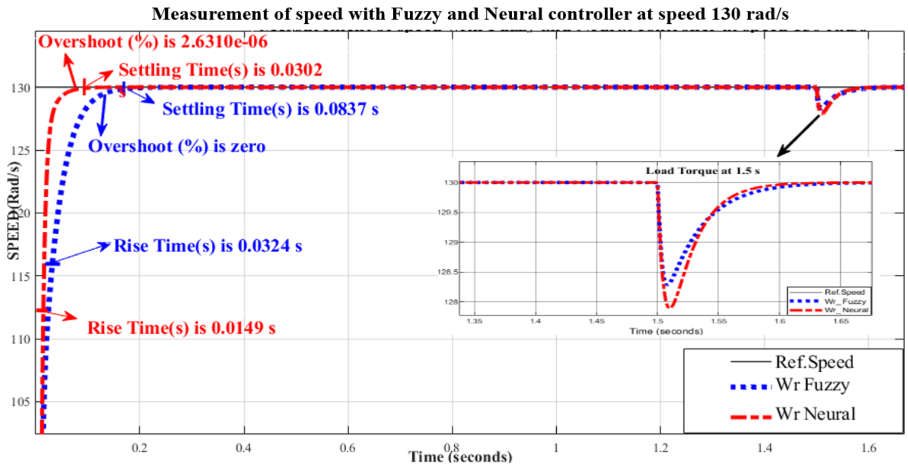


Fig. 12 Speed Responses of PMDC motor to step speed reference of 130 rad/s

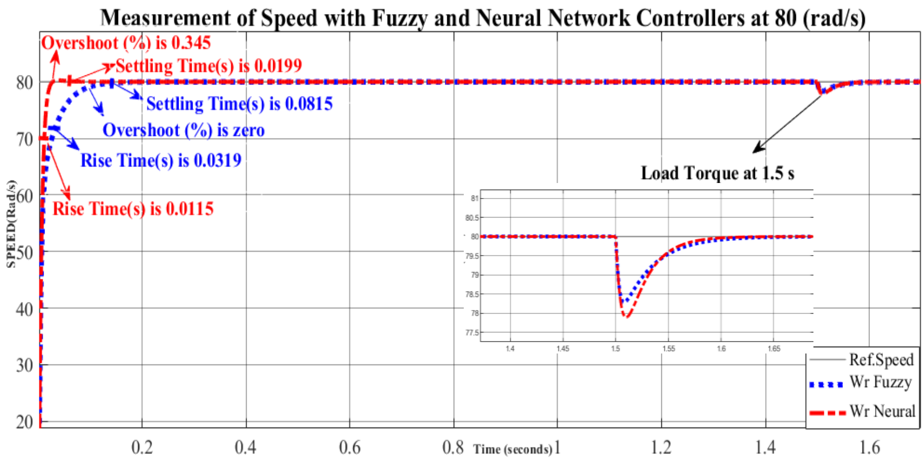


Fig. 13 Speed Responses of PMDC motor to step speed reference of 80 rad/s

At both speeds, the neural network controller consistently outperforms the fuzzy controller. The reaction times for both controllers at each speed are compiled in Table 4. According to this data, the neural controller is better suited for applications needing quick reactions since it responds to input changes and stabilizes the system more rapidly than the fuzzy controller. The neural network controller (red) reaches the reference speed more rapidly than the fuzzy controller (blue).

Table 4 Comparison of Response Times (in seconds)

Speed (rad/s)	Controller	Rise Time(s)	Settling Time(s)	Overshoot (%)	Peak Time (s)	Steady-State Error (EssE)
130	Neural	0.0149	0.0302	2.6310e-06 (Negligible)	2.1356	2.6310e-06
	Fuzzy	0.0324	0.0837	0.0000	0.9460	1.9677e-13
80	Neural	0.0115	0.0199	0.3455%	0.0374	4.2753e-06
	Fuzzy	0.0319	0.0815	0.0000	0.9490	1.5987e-13

Overshoot is a critical parameter in control systems, as excessive overshoot can lead to instability. The fuzzy controller eliminates overshoot at both speeds, while the neural controller exhibits a minimal overshoot that can be neglected, as shown in Table 4. At 80 rad/s, the neural controller reaches its peak in 0.0374 seconds, while the fuzzy controller requires a significantly longer duration of 0.9490 seconds. Conversely, at 130 rad/s, the fuzzy controller attains its peak earlier at 0.9460 seconds, whereas the neural controller takes 2.1356 seconds. This information indicates that at higher speeds, the fuzzy controller reaches its maximum output more quickly than the neural controller. Figs. 12 and 13 reinforce this data, showing how the fuzzy controller maintains a smooth response, while the neural controller exhibits a small but negligible overshoot.

Steady-state error (EssE) measures how accurately a controller maintains the desired speed over time. The fuzzy controller maintains much lower steady-state errors at both speeds, making it more suitable for applications requiring high precision. The figures further confirm this, showing how the fuzzy controller settles at the reference speed with minimal deviation. To assess how well each controller maintains speed stability under disturbances, a sudden increase in load torque at 1.5 s was applied. The neural controller returns to the reference speed faster than the fuzzy controller. However, at the moment of additional load application, the neural controller exhibits a sharp drop in speed, whereas the fuzzy controller shows a smoother and more stable response. The results are summarized in Table 5.

Table 5 Load Disturbance Response

Controller	Speed Drop (%)	Recovery Time (s)
Fuzzy (FLC)	1.3%	0.86 s
Neural (NNC)	1.6%	0.27 s

The fuzzy logic controller (FLC) experiences a smaller speed drop of 1.3% compared to the neural network controller (NNC), which drops by 1.6%, demonstrating better resistance to disturbances. However, the NNC recovers significantly faster, taking only 0.27 seconds to stabilize, whereas the FLC requires 0.86 seconds, making the NNC more responsive. Utilizing performance data from Tables 4 and 5, the following summary in Table 6 gives an overall comparison of the Fuzzy Logic Controllers (FLC) and the Artificial Neural Network (ANN) controllers, on a performance basis:

Table 6 Comparison Summary: FLC vs. ANN

Performance Metric	Comparative Analysis
Response Time	ANN has 50-60% quicker reaction times than FLC at all evaluated speeds, making it ideal for applications that demand rapid system response.
Overshoot	FLC has flawless overshoot performance, while ANN has very little overshoot, which is fine for most uses but might be a problem for precision systems.
Robustness	FLC has superior steady-state accuracy and a slightly better ability to reject disturbances, while ANN recovers from disturbances around three times quicker.
Complexity	FLC usually has a control structure that is easier to understand and see through, whereas ANN works more like a "black box" but can simulate more complicated interactions without needing to be programmed.
Implementation Cost	FLC is typically more efficient in resource-constrained systems, whereas ANN may demand more computing resources yet perform better in complex, data-rich contexts.

To determine robustness, PMDC motor parameters were varied within ±30% of their values and simulated using both FLC and ANN controllers. The new parameters changed to (RA=5.5141, L=0.0044, J=0.0022, B=0.0016, KE=0.6304). Performance parameters, including overshoot, settling time, and steady-state error (EssE), were identified. The ANN controller shows higher resilience to parameter fluctuations. Table 7 shows extensive results.

Table 7 Comparison of Response Times (in seconds)

Speed (rad/s)	Controller	Rise Time(s)	Settling Time(s)	Overshoot (%)	Peak Time (s)	Steady-State Error (EssE)
130	Neural	0.0170	0.0313	2.6310e-06	0.6052	2.6310e-06
	Fuzzy	0.0329	0.0840	0.0000	0.9374	1.7490e-13
80	Neural	0.0131	1.5217	0.8743	0.0376	4.2753e-06
	Fuzzy	0.0324	1.5129	0.0000	0.9384	1.5987e-13

Though neural network controllers (NNCs) react more quickly than fuzzy logic controllers (FLCs), the FLCs can retain their stability under disturbance. FLCs are computationally simple and interpretable, requiring tuning only at the beginning of their utilization. Fuzzy logic controllers also do not accommodate adaptations, while NNCs require a significant amount of computing to train, but become very adaptable to changes in the system once trained. Therefore, FLCs are useful in clearly defined and interpreted systems, while NNCs are useful when the system is dynamic, and humans must adjust to continuously changing inputs.

In either case, a hybrid control approach can exploit the best of both worlds, with NNCs reacting quickly and FLCs producing accurate and robust performance. Although simulation results showed the applicability of both ANN and FLC for the control of PMDC motors, it is important to account for potential practical limitations. FLCs generally consume fewer computing resources, allowing control on less capable microcontrollers, such as ATmega2560-based Arduino Mega or STM32; a very practical consideration for projects with limited computing abilities and power.

One of the major difficulties in deploying artificial/neural networks (ANNs) and fuzzy logic controllers (FLCs) to embedded/digital systems is the limited processing ability and memory resources that are often available, as well as the strict time constraints imposed by the real-time control process. The use of ANN architectures often requires a certain measure of tuning or optimization (e.g., quantization) to provide effective and efficient execution with the processing and memory resources available on the digital processor. Depending on the complexity of the FLC, it can be difficult to determine the actual resource usage.

In terms of achieving an implementation of either ANN or FLC for real-time production, there will also be increased resource constraints to contend with, including elements such as power consumption, interface latency, and system lag, in addition to limitations with respect to debugging and testing. Because the system must also be robust and tolerant to noise as well as able to achieve reliable real-time implementations, additional care will need to be given to tuning of parameters, simplification of models, and opportunities for using ancillary tools such as auto-code generation and hardware-in-the-loop testing strategies.

9. Conclusions

This research presents a comparative analysis between artificial neural networks (ANN) and fuzzy logic controllers (FLCs) in controlling the speed of a Permanent Magnet DC (PMDC) motor. These controllers were developed and implemented in MATLAB/Simulink, and then their performances were compared based on rise time, settling time, overshoot, speed drop, and recovery time. The purpose was performance evaluations under various conditions of operation. The findings include:

(1) The ANN controller exhibited a response speed that was far superior to that exhibited by the FLC. The ANN was able to reach its desired speed in just 0.0149 seconds (rise time) and stabilize within 0.0302 seconds (settling time) when set to a reference speed of 130 rad/s.

(2) The ANN controller exhibited negligible overshoot at 130 rad/s (2.63×10^{-6} %), while FLC had zero overshoot. At 80 rad/s, ANN displayed a small overshoot of 0.3455%, whereas FLC maintained zero overshoot. This indicates that FLC provides inherently stable operation, whereas ANN may introduce slight transient fluctuations.

(3) The recovery time from a speed drop for the ANN controller was 0.27s, whereas for the FLC it was 0.86 s demonstrating faster recovery for the ANN. This shows that FLC performs better concerning persistent control errors and disturbance resistance.

(4) The ANN controller attained peak speed values in closer proximity to the reference with the minimal discrepancy, more so at lower speeds. Nonetheless, at 130 rad/s, the peak time for ANN was 2.1356 s, while for FLC it was 0.9460 s, which evidences that FLC reaches steady-state stability faster after the initial transients.

(5) ANN controllers offer greater adaptability and dynamic performance. In contrast, FLC provides a controlled, rule-based approach that is advantageous for its low computational demand and easy implementation.

The selection between ANN and FLC is contingent upon the requirements for the given application. While the ANN controller is preferred for applications that require adaptability and rapid response, FLCs are a more desirable option when reliability, stability, and interpretability are required in cases of predetermined working conditions. Future work will involve hybrid ANN-FLC controllers to capitalize on the advantages of both approaches. Additionally, hardware setup would provide many insights into the real-world implementation and feasibility of controller performance under practical constraints.

Conflicts of Interest

The authors declare no conflict of interest.

References

[1] W. Aribowo, S. Muslim, F. Achmad, and A. C. Hermawan, "Improving Neural Network Based on Seagull Optimization Algorithm for Controlling DC Motor," J. Elektronika dan Telekomunikasi (JET), vol. 21, no. 1, pp. 48-54, 2021.

[2] N. L. Manuel, N. İnanç, and M. Lüy, "Control and Performance Analyses of a DC motor using Optimized PIDs and Fuzzy Logic Controller," Results in Control and Optimization, vol. 13, article no.100306, 2023.

[3] A. G. Abdullah, M. A. Ibrahim, and A. S. Saleh, "Performance Enhancing Speed Control of a DC Motor Based on an Artificial Neural Network," Journal Européen des Systèmes Automatisés, vol. 57, no. 5, pp. 1489-1495, 2024.

[4] A. K. Maidanov, S. K. Atanov, H. Canbolat, and S. A. Aljawarneh, "Advanced Control Strategies for DC Motor Speed Regulation: A Comparative Study of Artificial Neural Networks and ANFIS," Proceedings of 8th International Conference on Digital Technologies in Education, Science and Industry (DTESI 2023), 2023.

[5] M. S. Mauludin, A. Nugroho, A. Hidayat, and S. D. Prasetyo, "Simulation of AI-Based PID Controllers on DC Machines Using the MATLAB Application," Journal Européen des Systèmes Automatisés, vol. 57, no. 1, pp. 281-287, 2024.

[6] N. Hamouda, B. Babes, A. Boutaghane, S. Kahla, and M. Mezaache, "Optimal Tuning of $PI^{\lambda}D^{\mu}$ Controller for PMDC Motor Speed Control Using Ant Colony Optimization Algorithm for Enhancing Robustness of WFSs," Proceedings of IEEE 1st International Conference on Communications, Control Systems and Signal Processing (CCSSP), pp. 364-369, 2020.

[7] U. Alset, A. Apte, and H. Mehta, "Implementation of Fuzzy Logic Based High Performance Speed Control System for PMDC Motor using ATMEGA-328P-PU Micro-Controller," Proceedings of IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT), pp. 1-5, 2020.

[8] W. G. Soliman, B. K. Priya, D. A. Reddy, P. V. S. Anusha, and D. V. R. K. Reddy, "Reconfigurable Microarchitecture-Based PMDC Prototype Development for IoT Edge Computing Utilization," IEEE Sensor Journal, vol. 21, no. 2, pp. 2334-2345, 2021.

[9] S. A. Deraz and F. Alharbi, "An Interior Point Algorithm for Parameter Estimation of a Geared PMDC Motor Using Current and Speed Step Responses," IEEE Access, vol. 11, pp. 121892-121901, 2023.

[10] E. Çelik and H. Gör, "Enhanced Speed Control of a DC Servo System Using PI + DF Controller Tuned by Stochastic Fractal Search Technique," Journal of the Franklin Institute, vol. 356, no. 3, pp. 1333-1359, 2019.

[11] A. K. Paul, "Robust PMDC Motor Control for Accurate Wire Feeding in GMAW Using Back EMF," IEEE Transactions on Industrial Electronics, vol. 67, no. 1, pp. 413-420, 2020.

[12] A. S. Poznyak, A. V. Nazin, and H. Alazki, “Integral Sliding Mode Convex Optimization in Uncertain Lagrangian Systems Driven by PMDC Motors: Averaged Subgradient Approach,” IEEE Transactions on Automatic Control, vol. 66, no. 9, pp. 4267-4273, 2021.

[13] H. Velasco-muñoz, J. E. Candelo-Becerra, F. E. Hoyos, and A. Rincón, “Speed Regulation of a Permanent Magnet DC Motor with Sliding Mode Control Based on Washout Filter,” Symmetry, vol. 14, no. 4, article no. 728, 2022.

[14] X. Zhang, “Design and Implementation of Fuzzy PID DC Motor Control System Based on STM32,” Proceedings of IEEE International Conference on Control, Electronics and Computer Technology (ICCECT, pp. 1129-1131, 2023.

[15] E. Çelik and N. Öztürk, “A New Fuzzy Logic Estimator for Reduction of Commutation Current Pulsation in Brushless DC Motor Drives with Three-Phase Excitation,” Neural Computing and Applications, vol. 31, no. 2 supplement, pp. 1125-1134, 2019.

[16] Y. Cai, “Artificial Intelligence Applied to Control of DC Drives,” Proceedings of IEEE 5th International Conference on Mechatronics Technology and Intelligent Manufacturing (ICMTIM), pp. 598-609, 2024.

[17] R. S. Debes and T. Kara, “Design and Simulation of a PID Neural Network Controller for PMDC Motor Speed and Position Control,” European Journal of Science and Technology, Special Issue 44, pp. 46-50, 2022.

[18] A. J. Humaidi and I. K. Ibraheem, “Speed Control of Permanent Magnet DC Motor with Friction and Measurement Noise using Novel Nonlinear Extended State Observer-Based Anti-Disturbance Control,” Energies, vol. 12, no. 9, article no. 1651, 2019.

[19] M. K. Kaloi, E. Ali, G. Mustafa, N. Nizamani, M. A. Memon, A. R. Khatri, et al., “Fuzzy-PID Based Control Scheme for PMDC Series Motor Speed Control,” Indian Journal of Science and Technology, vol. 13, no. 28, pp. 2911-2923, 2020.

[20] H. M. Htun, A. N. Yakunin, H. S. Paing, and K. Win, “Implementation of ANFIS Controller for DC Motor on an Arduino Due Board,” Proceedings of IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus), pp. 2668-2671, 2021.

[21] E. Çelik, G. Bal, N. Öztürk, E. Bekiroglu, E. H. Houssein, C. Ocak, et al., “Improving Speed Control Characteristics of PMDC Motor Drives using Nonlinear PI Control,” Neural Computing and Applications, vol. 36, no. 16, pp. 9113-9124, 2024.

[22] A. Altay and A. B. Yildiz, “Complete Electrical Equivalent Circuit Based Modeling and Analysis of Permanent Magnet Direct Current (DC) Motors,” WSEAS Transactions on Circuits and Systems, vol. 21, pp. 182-187, 2022.

[23] A. M. Almawla, M. J. Hussein, and A. T. Abdullah, “A Comparative Study of DC Motor Speed Control Techniques Using Fuzzy, SMC and PID,” Journal Européen des Systèmes Automatisés, vol. 57, no. 2, pp. 397-406, 2024.

[24] K. H. Reddy, S. Sharma, A. Prasad, and S. Krishna, “Neural Network Based Fuzzy Plus PD Controller for Speed Control in Electric Vehicles,” Proceedings of IEEE International Conference on Interdisciplinary Approaches in Technology Management for Social Innovation (IATMSI), pp. 1-4, 2024.

[25] A. M. Ali, A. H. Mohammed, and H. M. Alwan, “Tuning PID Controllers for DC Motor by Using Microcomputer,” International Journal of Applied Engineering Research, vol. 14, no. 1, pp. 202-206, 2019.

[26] E. Eker, M. Kayri, S. Ekinci, and D. Izci, “A New Fusion of ASO with SA Algorithm and Its Applications to MLP Training and DC Motor Speed Control,” Arabian Journal for Science and Engineering, vol. 46, no. 4, pp. 3889-3911, 2021.

[27] U. Y. Poornima, M. Radhika, N. P. Spoorti, U. T. Shaikh, and A. B. Raju, “Separately Excited DC Motor Speed Estimation,” Proceedings of IEEE 5th India Council International Subsections Conference (INDISCON), Chandigarh, India, pp. 1-5, 2024.

[28] K. Patan and M. Patan, “Neural-Network-Based Iterative Learning Control of Nonlinear Systems,” ISA Transactions, vol. 98, pp. 445-453, 2020.