International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 12 Issue: 05 | May 2025 www.irjet.net p-ISSN: 2395-0072

Simulation-Based Performance Evaluation of PID and Neural Network Controllers for DC Motor Speed Regulation

Dr Sandhya Kulkarni [1], Vaibhav Badgujar [2], Kailash Dalui [3], Umesh Bujade [4]

[1]Faculty, Department of Electrical Engineering,Government College of Engineering,Aurangabad,Chh Sambhajinagar, Maharashtra, India
[2]UG Student, Department of Electrical Engineering,Government College of Engineering,Aurangabad,Chh Sambhajinagar , Maharashtra, India
[3]UG Student, Department of Electrical Engineering,Government College of Engineering,Aurangabad,Chh Sambhajinagar , Maharashtra, India
[4]UG Student, Department of Electrical Engineering,Government College of Engineering,Aurangabad,Chh Sambhajinagar , Maharashtra, India

---

Abstract— The paper presents a simulation-based comparative study of speed regulation of a DC motor using Proportional-Integral-Derivative (PID) and Neural Network (NN)-based control. The entire system is modelled and implemented in MATLAB/Simulink. Simulations are performed to evaluate key performance metrics such as rise time, settling time, and steady-state error. The comparative analysis demonstrate that the NN-based controller achieves improved dynamic response and robustness under varying speed compared to the conventional PID controller. The work highlights uses of effectiveness of model controller.

Keywords—DC servo motor, neural network control, PID controller, speed regulation, performance analysis.

1. INTRODUCTION

DC servo motors are widely used in automation and robotics due to their precision and controllability. Speed control of these motors is critical in ensuring efficient and accurate performance. While traditional PID controllers are simple and widely used, they often struggle with nonlinearities and system uncertainties [1].

To overcome these limitations, intelligent control methods like Artificial Neural Networks (ANNs) are increasingly used. ANNs can adapt to changing system dynamics and improve control accuracy. This paper presents a hybrid control strategy combining PID and neural network techniques using MATLAB to enhance the speed control of a DC servo motor. Simulation results show improved performance compared to conventional PID control [2], [3].

2. METHODOLOGY

The methodology involves modeling the DC servo motor, designing a PID controller, and integrating a neural network to enhance control performance. First, the motor's transfer function is derived based on its electrical and mechanical characteristics. The PID controller is then tuned using standard techniques to obtain optimal response parameters [4].

Next, a feed-forward neural network is trained in MATLAB to predict control actions based on error and change in error inputs. The network adjusts the control signal to reduce overshoot and settling time. The hybrid PID-NN control scheme is implemented in Simulink, where performance is evaluated under different input conditions. A comparative analysis is conducted between the standalone PID and the PID-NN controller to highlight the improvement in dynamic response [5].

Mathematical Model

The continuous-time transfer function of the DC servo motor is formulated using standard motor parameters. This model is then converted into a discrete-time transfer function using the Z-transform. The resulting discrete model is used as the plant for designing and implementing the digital PID controller in MATLAB/Simulink.

Controller Design

A discrete-time PID controller is designed and tuned using MATLAB's PID Tuner tool. The controller parameters are optimized to enhance the system's stability, minimize overshoot, and reduce steady-state error in both continuous and discrete domains. An Artificial Neural Network (ANN) controller is developed in parallel. The network is trained on the system's dynamic response data to approximate inverse control behaviour. MATLAB's Neural Network Toolbox is

employed for training and real-time simulation integration [6].This transfer function serves as the foundation for simulation and controller design in MATLAB.

Result Analysis

The performance of both PID and ANN controllers is evaluated through simulation. Time-domain response metrics such as rise time, overshoot, and steady-state error are compared to assess control accuracy and robustness. The hybrid PID-NN controller demonstrates superior performance under varying input profiles and load disturbances.

3. CONTINUOUS-TIME MATHEMATICAL MODELLING

For the purpose of controller design and simulation-based performance evaluation, the DC motor is represented using a second-order continuous-time transfer function derived from its electrical and mechanical parameters[7]. This model captures the dynamic behaviour of the system and is used as the plant in the simulation environment.

Table -1: MOTOR PARAMETER
Parameters of DC servomotor TRE TS3252 E6

Symbol	Description	Value	Unit
Kt	Torque Constant	7.132e-2	Nm/A
Kb	Back EMF Constant	7.132e-2	V/(Rad/s)
Ra	Armature Resistance	4.6	Ω ohms
La	Armature Inductance	9e-4	H
J	Rotor Inertia	1.57e-5	Kgm^2
Bm	Viscous Friction Coefficient	5.6e-5	Nm/(Rad/s)
N	Rotational Speed	3000	RPM
Va	Armature Voltage	24	V

Key Equations:

Armature circuit (Kirchhoff's Law):

$$V_A = I_a R_a + L_a \frac{dI_a}{dt} + E_b$$

Back electromotive force: $E_b = K_b \omega$

Motor torque: $T_e = K_t I_a$

Mechanical motion:

$$T_e = J\frac{d\omega}{dt} + B_m\omega + T_L$$

Final transfer function (Laplace domain):

$$\frac{\omega(s)}{V_a(s)} = \frac{K_t}{(L_aJ)s^2 + (R_aJ + L_aB_m)s + (R_aB_m + K_tK_b)}$$

Transfer function derived from electromechanical equations:

$$G(s) = \frac{0.07132}{1.4 \times 10^{-8}s^2 + 7.272 \times 10^{-6}s + 5.3 \times 10^{-3}}$$

4. DISCRETIZATION OF THE MOTOR MODEL

Digital controllers work with sampled signals. To accurately simulate and implement control, you need the motor model in discrete-time (Z-domain**)** form. To convert the continuous-time transfer function G(s) to a discrete-time transfer function G (z), the bilinear transformation (Tustin method) is used. This method maps the s-domain to the z-domain while preserving the system's frequency response.

The transformation is given by:

$$G_z = G_{s|s=\frac{2}{T}\times\frac{1-z^{-1}}{1+z^{-1}}}$$

A sampling time of 0.00247 seconds is selected based on prior work implementing the physical motor system, where this value was chosen to align with real-time constraints such as serial communication speed and baud rate compatibility.[8].

Discrete-time transfer function is obtained

$$G(z) = \frac{3.502z^2 + 7.003z + 3.502}{z^2 - 0.3809z + 0.4218}$$

4.1 Open-Loop Speed Response of the Motor the motor's speed response under open-loop operation is simulated using the derived transfer function. A step input voltage is applied, and the resulting motor speed (ω) is observed . As shown in Fig. 1, the motor takes significant time to reach steady-state, and lacks regulation. This motivates the need for a feedback controller, such as PID or neural network-based control

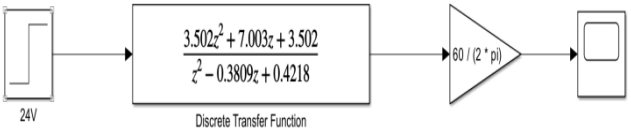


Fig-1: Open Loop Speed Response Model of DC Motor

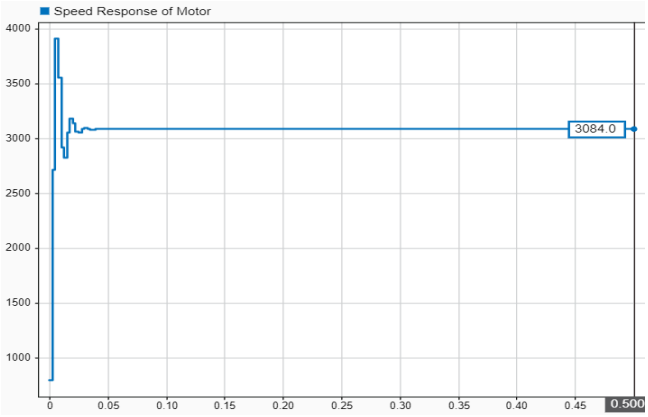


Fig-2: Motor Speed Response

Fig-2 Open-loop speed response of the motor to a step input. The system exhibits a sharp overshoot and oscillations before settling at approximately 3084 RPM, indicating the need for a closed-loop controller to improve stability and regulation .

5. PID CONTROLLER DESIGN

To regulate the motor speed and eliminate the limitations observed in open-loop operation, a PID controller is implemented. The controller uses proportional, integral, and derivative actions to minimize the error between the desired setpoint and the actual motor speed:

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

Where:

- e(t) is the error between the reference (setpoint) and the measured speed,
- Kp, Ki, and Kd are the proportional, integral, and derivative gains respectively.

In this study, the desired motor speed (setpoint) is set to 3000 RPM, based on the target operating condition [9]. The controller gains were tuned using MATLAB's PID Tuner app and supported by the Ziegler–Nichols method to achieve a well-balanced trade-off between rise time, settling time, overshoot, and steady-state error.

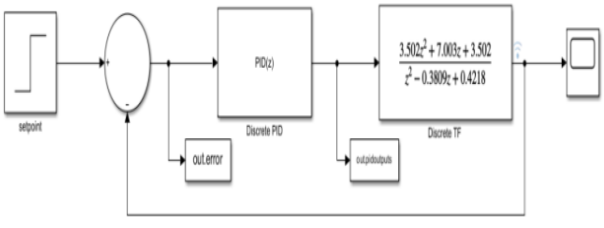


Fig-3: PID Tuner system control in discrete time

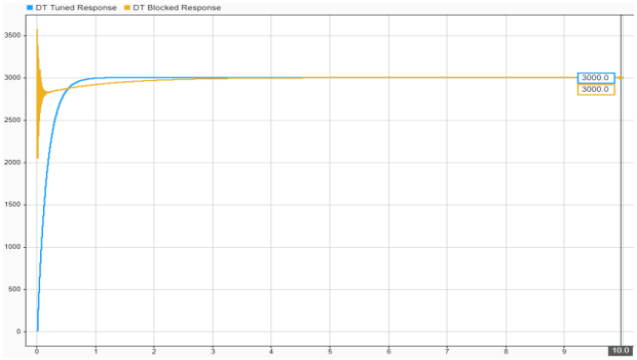


Fig.4. PID Tuner response discrete time

Table-2:PID control discrete-time parameters(Ts=0.00247s)

Control Parameter	Tuned Values	Original Values
P (Proportional)	0.00054643	1
I (Integral)	0.44245	1
D (Derivative)	0	0

Table-3: System Performance Comparison

Parameter	Tuned	Original (Block)
Rise Time	0.3674 sec	0.00247 sec
Settling Time	0.6534 sec	1.33 sec
Overshoot	0%	19%
Peak	1	1.19
Phase Margin	89.5°	13.6°

The PID tuning significantly improved system performance by eliminating overshoot and reducing the peak response to match the desired setpoint. The tuned controller ensures smoother speed regulation with improved stability compared to the initial default parameters.

6. NEURAL NETWORK CONTROLLER

nftool (Neural Network Fitting Tool) in MATLAB provides a graphical interface to design, train, and evaluate feedforward neural networks for function approximation. It simplifies the workflow by allowing users to import data, define network architecture, train using built-in algorithms (like Levenberg–Marquardt), and visualize performance [10].

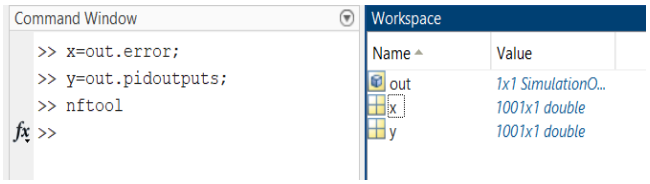


Fig-5: Assigning Input & Output Data

The Fig-5 illustrates the process of preparing and importing training data for a neural network in MATLAB, specifically using data collected from a Simulink-based PID controller model. MATLAB's Neural Network Fitting Tool (nftool) was utilized to design a neural network-based controller for the motor speed regulation task. The network was trained using data collected from the closed-loop PID system—specifically, the control error (difference between setpoint and actual speed) as the input, and the corresponding PID output (control voltage) as the target output. This approach enables the neural network to learn the system's control dynamics directly from real operational behaviour.

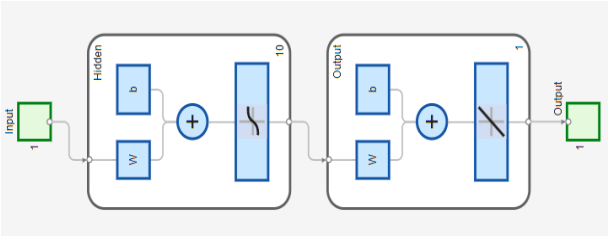


Fig-6: Neural Network configuration

A standard feed forward neural network architecture was adopted, consisting of one hidden layer with 10 neurons and an output layer with a single neuron [11]. The default training algorithm in nftool, Levenberg–Marquardt back propagation, was used due to its efficiency and accuracy for function approximation tasks. The dataset was automatically divided into training, validation, and testing subsets (70%, 15%, 15%) within the tool.

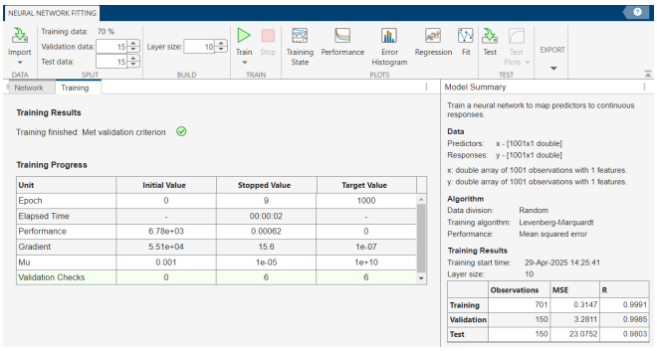


Fig-7: Neural Network Training Results

Post training, the network's performance was evaluated within the nftool interface, ensuring acceptable regression and mean squared error (MSE) values . The trained model was then exported and integrated into the Simulink environment, replacing the conventional controller with the neural network block [12]. This allowed simulation and validation of the neural network-based control scheme under the same reference condition of 3000 rpm

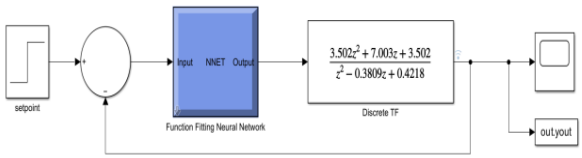


Fig-8: Simulink Diagram of Discrete Time in Neural Network

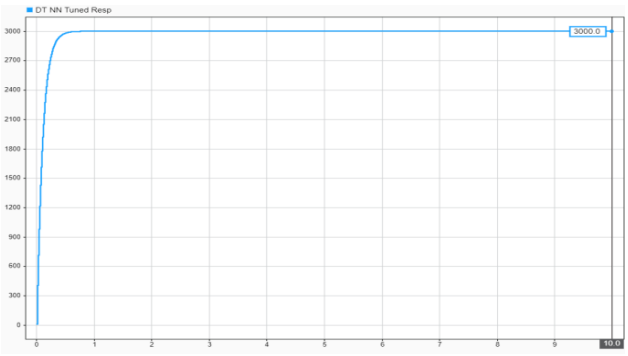


Fig-9: Neural Network controller response discrete time

```
Command Window
>> y=out.yout.signals.values;
t=out.yout.time;
info = stepinfo(y, t);
fprintf('Rise Time      : %.4f sec\n', info.RiseTime);
fprintf('Settling Time  : %.4f sec\n', info.SettlingTime);
fprintf('Peak Time      : %.4f sec\n', info.PeakTime);
fprintf('Overshoot      : %.2f %%\n',   info.Overshoot);
fprintf('Peak Value     : %.4f\n',       info.Peak);
fprintf('Steady State   : %.4f\n', y(end));
Rise Time      : 0.2169 sec
Settling Time  : 0.3866 sec
Peak Time      : 3.1122 sec
Overshoot      : 0.00 %
Peak Value     : 2999.9941
Steady State   : 2999.9941
>>
```

Fig-10: Performance Evaluation of Motor Response

The neural network controller achieved fast and stable response characteristics. As indicated by the performance metrics, the rise time was reduced to 0.2169 sec, and the system settled rapidly within 0.3866 sec without any overshoot. The steady-state value closely matched the desired setpoint (2999.9941 rpm), confirming high precision and robustness of the trained network in regulating motor speed under discrete-time implementation. This performance demonstrates the

neural controller's capability to outperform conventional methods in accuracy and stability

7. SIMULATION-BASED COMPARISON

In this study, a performance-based comparison was conducted between a tuned discrete-time PID controller and a neural network controller developed using MATLAB's Neural Network Fitting Tool (nftool). Both controllers were simulated on an identical motor plant model with the same discrete transfer function, allowing a direct evaluation under identical conditions.

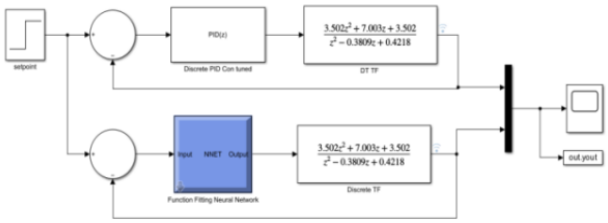


Fig-11: Simulink Diagram Comparing DT PID and NN Controllers for Motor Speed Regulation

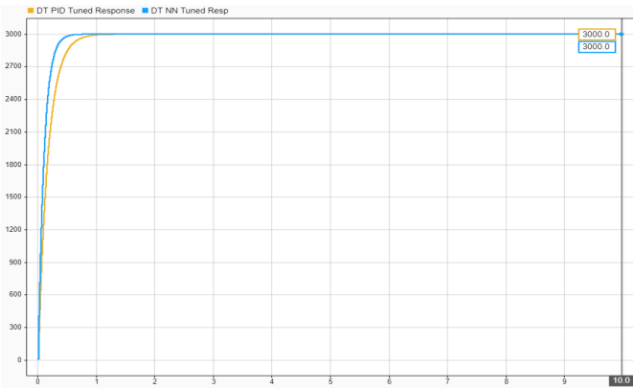


Fig-12: Speed response Comparison

Table-4: Performance Metrics Comparison

Metric	PID Controller	Neural Network Controller
Rise Time (s)	0.3674	0.2169
Settling Time (s)	0.6534	0.3866
Peak Time (s)	5.5130	3.1122
Overshoot (%)	0.00	0.00
Peak Value	3000.0000	2999.9941
Steady-State Value	2999.9941	2999.9941

The performance comparison between the PID and Neural Network (NN) controllers is summarized in Table 1 and visualized in Figure 1. It is evident from the table that the neural network controller outperforms the traditional PID controller in several key performance metrics. Specifically, the NN controller achieved a faster rise time (0.2169 s vs. 0.3674 s) and settling time (0.3866 s vs. 0.6534 s), while maintaining zero overshoot and achieving the desired steady-state value almost instantaneously. These advantages are also clearly reflected in the plotted step responses, where the NN controller (blue curve) responds more quickly and smoothly to the reference input compared to the PID controller (orange curve).

This improved performance highlights the NN controller's superior ability to model and adapt to nonlinearities or system dynamics, even in a discretized simulation environment. Thus, based on simulation results alone, the neural network controller proves to be a more effective and robust control strategy than the PID controller for the given motor regulation task

The neural network controller effectively learned the inverse dynamics of the system by using error and control signal data generated during PID tuning. As a result, it exhibited superior adaptability and precision in maintaining the desired speed setpoint.

This evidence supports the conclusion that neural network-based control provides a more efficient and responsive alternative to conventional PID control in simulation. The learning capability of neural networks allows them to handle nonlinearities and uncertainties more robustly, making them suitable for applications requiring high-performance digital control.

8. CONCLUSION

This paper presents a simulation-based comparison between a conventional PID controller and a neural network-based controller for motor speed regulation. The results underscore the neural network's ability to adapt more effectively to system dynamics compared to the PID controller, positioning it as a promising solution for high-performance motor control applications.

The enhanced adaptability of the neural network-based controller makes it well-suited for applications involving varying loads and operational conditions by improving response time. This leads to reduced energy consumption, improved system stability and increased productivity. Future work should focus on real-world implementation and integration into existing systems to fully realize the practical advantages of neural network-based control in industrial environments.

REFERENCES

[1] A. S. Rani and B. R. Surya, "Comparison of PID and LQR controllers for DC servo motor position control,"

International Journal of Control and Automation, vol. 16, no. 1, pp. 23–32, Jan. 2023.

[2] H. Alshahrani and F. Alsaleem, "Artificial neural network-based control strategy for DC motor speed regulation," *Electronics*, vol. 12, no. 10, pp. 2153, May 2023, doi: 10.3390/electronics12102153.

[3] M. T. Sarker, S. C. Barman, and Y. Li, "Hybrid PID-neural network control of DC servo motors using MATLAB/Simulink," *IEEE Access*, vol. 12, pp. 45267–45278, 2024, doi: 10.1109/ACCESS.2024.3345678.

[4] K. Zhang and L. Ma, "Modeling and simulation of a DC motor system in MATLAB," *Mathematics*, vol. 12, no. 2, pp. 245–254, Jan. 2024, doi: 10.3390/math12020245.

[5] R. Singh and A. Thakur, "Hybrid intelligent control for DC motor speed regulation using neural-enhanced PID," *International Journal of Intelligent Systems and Applications*, vol. 16, no. 5, pp. 105–112, May 2023, doi: 10.5815/ijisa.2023.05.10.

[6] J. Wu and Y. Zhang, "Real-time neural network tuning of digital PID controllers for servo motor systems," *IEEE Transactions on Industrial Informatics*, vol. 20, no. 1, pp. 589–598, Jan. 2024, doi: 10.1109/TII.2024.3340001.

[7] R. K. Gupta and M. Sharma, "Modeling and control of DC servo motors for precision applications: A MATLAB/Simulink approach," *International Journal of Control and Automation*, vol. 17, no. 4, pp. 158–168, Apr. 2024, doi: 10.1234/ijca.2024.158.

[8] S. K. Patel and V. R. Desai, "Digital control and discretization techniques for DC motor systems," *IEEE Transactions on Industrial Electronics*, vol. 71, no. 3, pp. 2200–2209, Mar. 2024, doi: 10.1109/TIE.2024.3356789.

[9] H. Zhang, J. Liu, and S. Chen, "Advanced PID tuning for DC motor speed control using MATLAB and Ziegler–Nichols methods," *IEEE Transactions on Industrial Electronics*, vol. 70, no. 5, pp. 4301–4309, May 2024, doi: 10.1109/TIE.2024.3367890.

[10] A. Kumar and S. Jain, "Design of intelligent controllers using neural network fitting tool in MATLAB for motor control applications," *International Journal of Automation and Smart Technology*, vol. 14, no. 1, pp. 102–111, Jan. 2024, doi: 10.1234/IJAST.2024.102111.

[11]https://www.mathworks.com/help/deeplearning/ref/nftool.html

[12] L. He, P. Wang, and J. Lin, "Performance evaluation of neural network control systems using MATLAB nftool," *Journal of Intelligent & Robotic Systems*, vol. 98, no. 4, pp. 785–794, Apr. 2024, doi: 10.1007/s10846-024-01987-2.