Iğdır Üniversitesi Fen Bilimleri Enstitüsü Dergisi, 15(3), 872-888, 2025
Journal of the Institute of Science and Technology, 15(3), 872-888, 2025

ISSN: 2146-0574, eISSN: 2536-4618

Electrical Electronic Engineering **DOI: 10.21597/jist.1592544**

Research Article

Received: 28.11.2024 **Accepted: 03.03.2025**

To Cite: Demircan, B. & Yaren, T. (2025). Effect of Filtering Techniques on the Derivative Term in Fuzzy Logic Controller for DC Motor Position Control. *Journal of the Institute of Science and Technology*, 15(3), 872-888.

Effect of Filtering Techniques on the Derivative Term in Fuzzy Logic Controller for DC Motor Position Control

Batın DEMİRCAN[1*], Tuğçe YAREN[2]

Highlights:

- Real-time DC motor position control
- Improved error reduction achieved using the Second-Order Filtered Derivative technique compared to Transfer Function–Based Derivative
- Implementation of a Fuzzy Logic Controller with advanced derivative filtering for real-time DC motor position control

ABSTRACT:

Direct Current (DC) motors are fundamental components in various industrial and automation systems, valued for their precision and controllability. Traditional control methods, such as Proportional-Integral-Derivative (PID) controllers, often require robust mathematical models and are susceptible to performance degradation under non-ideal conditions. This study investigates the implementation of Fuzzy Logic Controllers (FLC) for real-time DC motor position control, with a focus on analyzing the impact of different derivative approaches. To construct a comprehensive mathematical model of the DC motor system, both white-box and black-box system identification approaches were employed. The white-box method utilized physical principles of the motor, while the black-box method relied on empirical input-output data. The Transfer Function-Based Derivative (TFD) and Second-Order Filtered Derivative (SOFD) techniques are evaluated for their maintaining system responsiveness. A test setup utilizing an STM32F4 discovery kit was developed, and the performance of both derivative approaches was compared using a repeating stair sequence as the reference input. The experimental results showed that both techniques performed successfully, but the SOFD method demonstrated a more effective error reduction. The findings offer insights into derivative filtering techniques, highlighting the benefits of incorporating advanced filtering strategies in FLC-based control systems.

Keywords:

- Dc motor
- Fuzzy logic control
- Derivative filter
- System identification
- Control Theory

[1] Batın DEMİRCAN (**Orcid ID:** 0000-0002-0765-458X), Balikesir University, Balikesir Vocational School, Department of Electronics and Automation, Balikesir, Türkiye
[2] Tuğçe YAREN (**Orcid ID:** 0000-0001-9937-3111), Balikesir University, Faculty of Engineering, Department of Electrical and Electronics Engineering, Balikesir, Türkiye
***Corresponding Author:** Batın DEMİRCAN, e-mail: batindemircan@gmail.com

INTRODUCTION

Proportional-Integral-Derivative (PID) control is the most widely used control method in industrial applications due to its simple and effective structure. The PID control strategy has become a practical choice in many Direct Current (DC) motor applications owing to its ease of implementation and tunability (Al-Bargothi, Qaryouti, & Jaber, 2019). However, traditional PID controllers may suffer from performance loss in nonlinear systems; therefore, adaptive PID controllers, which adjust parameters through techniques like fuzzy logic, can enhance robustness against uncertainties (Akbari-Hasanjani, Javadi, & Sabbaghi-Nadooshan, 2014; Gebremariam & Alemu, 2023). Additionally, the performance of PID controllers can be improved through the integration of optimization techniques, such as genetic algorithms, which enable precise tuning of control parameters (Ortatepe, 2023).

The mathematical model of a system plays a crucial role in the success of a PID controller. An accurate model facilitates the effective tuning of control parameters, aiding in achieving the desired performance. However, obtaining a mathematical model is particularly challenging for complex and nonlinear systems. Uncertainties in the modeling process and changes in the system's dynamic structure can negatively impact PID control performance. Fuzzy Logic Controllers (FLC) are effectively used in managing nonlinear and complex systems where obtaining an accurate mathematical model is difficult by integrating human-like reasoning into control strategies (Liu et al., 2025).

FLCs have demonstrated significant improvements in DC motor control applications compared to traditional methods (Sadi, 2020; Kroičs & Būmanis, 2024). The integration of fuzzy logic with PID control yields positive results, particularly in self-tuning applications where control parameters adapt to changing conditions in real time (Akbari-Hasanjani et al., 2014; Xue et al., 2012). This control structure is critically important for systems with varying load conditions, as it ensures sustained performance (Hidayati & Prasetyo, 2016). In addition to PID and FLC methods, various alternative approaches such as Linear Quadratic Regulator (LQR), Sliding Mode Control (SMC), and Model Predictive Control (MPC) are also widely utilized in DC motor control (Gu et al., 2015; Ma'arif & Çakan, 2021; Prasad et al., 2014; Shah et al., 2018). In a comparative study on DC motor speed control, it was reported that by employing rule-based fuzzy inference, the nonlinear characteristics of the motor were effectively addressed, resulting in smoother transient and steady-state responses under the FLC. Furthermore, the FLC significantly decreased overshoot and settling time and outperformed the PID approach (Raza et al., 2024).

Each of these methods has distinct characteristics that adapt to specific applications and operating conditions. For instance, SMC and Super-Twisting Algorithms offer high robustness in nonlinear systems and reduce the chattering effect, making them a suitable choice for DC motor control (Valenzuela et al., 2020). Similarly, the Extended Kalman Filter (EKF) and its improved variants such as the Improved Iterated Extended Kalman Filter (IIEKF) have been effectively utilized for self-tuning control and parameter estimation in DC motors (Moaveni et al., 2023; Saleem & Omer, 2017). EKF-based adaptive PI control schemes have shown promising results in optimizing motor speed control, especially in real-time applications requiring robustness against parameter variations.

The role of derivative terms in motor control applications is crucial, as it directly influences system stability and responsiveness. The derivative of the position or velocity signals is commonly used to predict future errors and enhance control accuracy. However, noise and disturbances can degrade the effectiveness of control by reducing the accuracy and stability of the derivative signals. To address this issue, filtering techniques such as low-pass filters and second-order filtered derivatives (SOFD) are employed to suppress noise while preserving essential system dynamics. By carefully selecting an

appropriate filtering technique, control precision can be significantly enhanced. This study investigates the effects of different derivative filtering techniques, providing insights into their influence on controller performance.

The STM32F4 microcontroller provides a suitable platform for DC motor control with high processing power and extensive hardware support. Its ability to perform high-speed processing and support advanced control algorithms significantly enhances motor control performance, simplifying application development. The Waijung block set seamlessly integrates with STM32F4, allowing for easy Simulink-based modeling, efficient code generation, and real-time execution, thereby simplifying the development of complex control algorithms. Due to these advantages, real-time DC motor position control applications in this study were developed using the STM32F4 development kit and Simulink models, implemented via the Waijung block set.

However, despite these advantages, the STM32F4 and the Waijung block set have certain limitations. While, Waijung facilitates Simulink-based modeling and code generation, its dependency on specific hardware and limited update frequency can create some constraints in terms of flexibility. Additionally, the STM32F4 series may be insufficient in processing capacity for more complex control systems that require high bandwidth.

Therefore, alternative microcontroller platforms with higher processing power and expandable hardware support can be considered depending on the application. For instance, the STM32H7 series, with its advanced processing power and higher clock frequency, offers improved performance in real-time control applications. DSP-based microcontrollers, such as the Texas Instruments TMS320F28379D, provide significant advantages in applications requiring high-speed and precise computations, such as motor control. Alternatively, FPGA-based systems and advanced embedded platforms like Raspberry Pi and Beagle Bone Black offer notable benefits in terms of flexibility and computational power. In addition to hardware alternatives, alternative software tools for Simulink-based code generation can also be considered. MATLAB Embedded Coder serves as a more versatile alternative, offering direct C code generation with broader hardware compatibility. Additionally, native development environments such as STM32CubeIDE or open-source solutions like Scilab/Xcos provide alternative approaches for embedded control system implementation, allowing for more customized and scalable solutions. The selection of an appropriate hardware platform and software should be based on the specific objectives of the study and the application requirements, considering the trade-offs between processing performance, hardware scalability, and implementation complexity.

In real-time DC motor position control applications, obtaining accurate speed data is critical. When dedicated sensors or hardware are unavailable, software-based approaches become essential. This research aims to enhance the performance of FLCs by examining the impact of derivative terms on system response. By incorporating Transfer Function-Based Derivative (TFD) and Second-Order Filtered Derivative (SOFD) techniques, we assess their effectiveness in noise suppression and control accuracy. The experimental outcomes identify the most effective derivative method for robust and efficient real-time DC motor control. To achieve this, both white-box and black-box modeling approaches were utilized to obtain the system transfer function. After analyzing the process of mathematical model development, an FLC was designed and implemented for real-time DC motor position control. The performance of the designed controller, including the Fuzzy Logic Controller with Transfer Function Derivative (TFD-FLC) and the Fuzzy Logic Controller with Second-Order Filtered Derivative (SOFD-FLC), was evaluated, allowing for a detailed comparison of their effects on system performance. Both designs demonstrated successful performance, and their performance characteristics were analyzed in detail. Moreover, this study highlighted the advantages of using model-independent

fuzzy approaches, overcoming the limitations of model-based control methods, and offered a new perspective that could serve as an alternative to traditional methods in the literature. Furthermore, the study guides those conducting experimental work, demonstrating that application development is feasible with a cost-effective test setup. Thus, it emphasizes that economical and efficient solutions can be achieved in both laboratory and industrial applications.

MATERIALS AND METHODS

System identification is a process aimed at understanding the behaviors of a physical system and converting these behaviors into a mathematical model. This process involves determining the input-output relationships of the system and creating mathematical representations of real-world systems. System identification plays a critical role, especially in engineering and control systems, in creating the mathematical models necessary for controlling and optimizing the system. These models make it possible to predict the behavior of a system, optimize control, and foresee future behaviors.

System identification can be performed using different approaches. Each approach offers advantages for different types of systems, and the choice of the correct method depends on the nature of the system, the data used, and the desired model accuracy. Two main approaches stand out: the white-box approach and the black-box approach. In the White-Box approach, complete knowledge of the system's internal structure and dynamics is utilized to create a model based on physical laws and engineering principles. This approach allows for an in-depth understanding and control of the system's behaviors; however, depending on the system's complexity, the modeling process can be challenging. In contrast, the Black-Box approach does not involve knowledge of the system's internal structure, and only input-output data are observed. Using this data, a mathematical model is created to predict the system's behaviors. The Black-box method is more suitable for cases where the physical structure of the system is unknown or complex and relies on data-driven analysis.

DC Motor System Identification Using White-Box Approach

White-box system identification involves the development of models based on the physical principles of the system. Therefore, this approach requires a comprehensive understanding of the electrical and mechanical properties of the motor, which facilitates the Equation of mathematical equations that define the system's dynamics. A white-box model of a DC motor includes armature voltage equations and the mechanical equations that govern the motor's motion, allowing for precise control and prediction of the motor's behavior under various conditions.

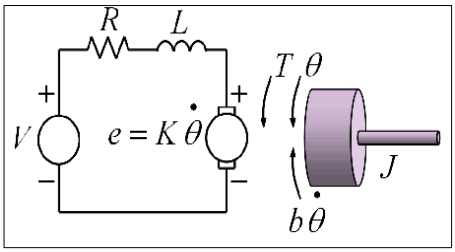

Figure 1. DC motor model

The white-box approach is particularly useful when accurate modeling is required for control system design, as it provides insights into the motor's internal workings and allows for the integration of various physical (Adukwu et al., 2023; Asadi, 2018). Additionally, this method can be implemented using simulation software tools like MATLAB/Simulink, which facilitate the analysis and validation of models with experimental data (Sami et al., 2021).In this study, the Canon FN38S model permanent magnet direct current motor was used.

In this study, the DC motor mathematical model will first be obtained using the white-box approach, which will utilize the motor's electrical and mechanical structure. A DC motor model is presented in Figure 1, and the variables and constants used in this model are provided in Table 1 along with their units and symbols.

When considering the electrical working structure of the DC motor, we conclude that the motor torque (T) is directly proportional to the armature current (i), and the counter-electromotive force (emf, e) is directly proportional to the angular speed (ω): $K_t = K_e = K$). The calculation of the value of "$T(t)$" is given in Equation 1, and the calculation of the value of "$e(t)$" is given in Equation 2.

$$T(t) = K_t i(t) \tag{1}$$

$$e(t) = K_e w(t) \tag{2}$$

According to Kirchhoff's voltage law, the equation has been obtained, as given in Equation 3.

$$V(t) = Ri(t) + L\frac{di(t)}{dt} + e(t) \tag{3}$$

Considering the mechanical structure, the motor torque equation can be expressed independently of current, as given in Equation 4.

$$T(t) = J\frac{d^2\theta(t)}{dt^2} + b\frac{d\theta(t)}{dt} \tag{4}$$

As can be seen, the equations considered are written in the time (t) domain. By taking the Laplace transforms of these equations, their forms in the "s" domain are written, and system transfer functions are derived from the voltage, back EMF, and torque equations. It is well known that the transfer function represents the relationship between the input and output of the system. The input to the motor is the voltage applied to the motor. The output, however, may vary depending on the application. In practice, the system output signals are the DC motor position and DC motor speed information. Therefore, two different transfer functions will be written as "motor position" and "motor speed" transfer functions. The transfer function representing the relationship between the motor's position and voltage is given in Equation 5.

$$\frac{\Theta(s)}{v(s)} = \frac{K}{s((Js+b)(Ls+R)+K^2)} \tag{5}$$

When the motor parameter values are substituted, the position transfer function of the motor used in the experimental setup is derived as shown in Equation 6.

$$\frac{\Theta(s)}{v(s)} = \frac{0.0338}{9.28e{-}09s^3+1.6e{-}05s^2+0.001142s} \tag{6}$$

Table 1. Dc motor parameters

Symbol	Definition	Value
J	Rotor Inertia	32 g-cm^2
B	Viscous Damping	-
K_i	Torque Constant	36.1 mN.m/A
K_m	Back EMF Constant	3.8 V/1000 r/1000

The motor parameters are taken from the motor's catalog. The catalog information does not include the viscous damping coefficient. For the white-box approach application, this parameter has been assumed to be zero.

DC Motor System Identification Using Black-Box Approach

Black-box modeling focuses on the system's input-output behavior without delving into the internal mechanisms of the system. This approach is particularly useful when the internal dynamics are complex or not fully understood. Black-box models are derived from empirical data obtained by

observing the system's response to specific inputs and are used to predict future behavior (Frances et al., 2019; Naziris et al., 2020).

To obtain the mathematical model of the motor, system identification was performed using the black-box model. The black-box approach is preferred when there is no information about the system's structure. In this method, the mathematical model of the system is created based solely on experimental data (Kizir et al., 2019). In the study, various input signals were applied to the system, and the system's output response was examined to develop a model. The input signal (voltage) applied to the system and the corresponding output signal (motor position) are shown in Figure 2a. Based on these input and output signals, a transfer function with two poles and one zero was obtained using the MATLAB System Identification Tool (SIT). The obtained transfer function is provided in Equation 7. The result graph generated from this transfer function is presented in Figure 2b. A similarity of 97.65% between the real system data and the transfer function output data indicates that the transfer function is appropriate for modeling the position of the DC motor.

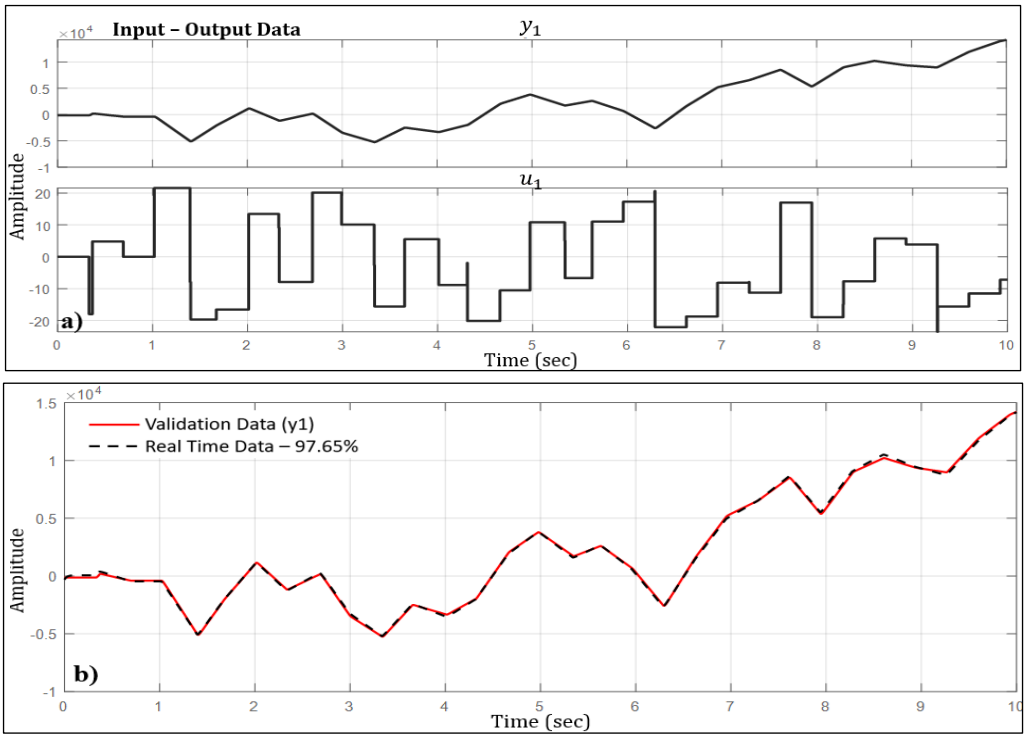

Figure 2. a) Input and output test signal used for system identification b) Result of system identification comparison

$$TF = \frac{778.6s + 523.3}{s^2 + 0.5973s + 0.1382} \tag{7}$$

Controller Design, Fuzzy Logic Controller

Deriving a mathematical model for a DC motor, whether through white-box or black-box system identification, can be a challenging and time-consuming process. It often involves detailed knowledge of the system dynamics, extensive data collection, and complex parameter estimation, which may still result in an incomplete or inaccurate model due to unmodeled nonlinearities or external disturbances. These limitations can impact the performance of model-based controllers. As an alternative, FLC eliminates the need for an exact mathematical model by employing a rule-based approach capable of handling uncertainties and nonlinearities.

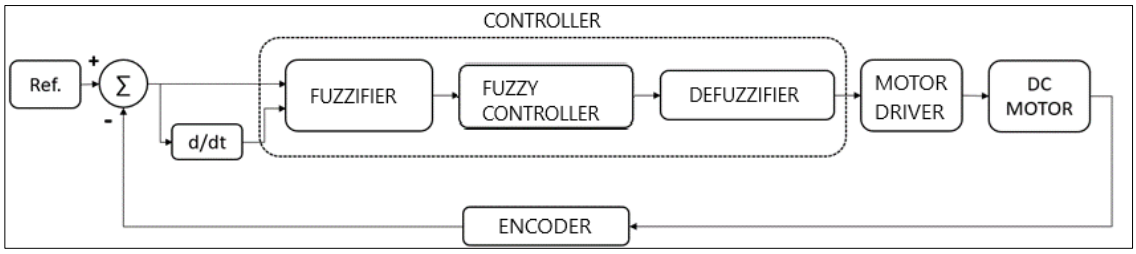

Figure 3. The block diagram of the FLC method

The block diagram of the FLC method is shown in Figure 3. For FLC, the first step is to calculate the error signal by taking the difference between the reference position and the instantaneous motor position read from the encoder. The derivative of the error signal is then taken to obtain information about the change in the error, and both the error and its rate of change are fuzzified. After this process, rule inference is performed based on the defined rule table, and the control signal is obtained through defuzzification.

The fuzzification process is carried out by generating membership functions for the required parameters. It is important to reference previously developed, tested, and validated membership functions to ensure accuracy. Applications performed with unvalidated membership functions may yield results that significantly deviate from the intended outcomes.

In this study, the membership functions used for fuzzifying the error and the rate of change of error are shown in Figure 4. Triangular functions were chosen as membership functions to reduce the computational load on the microcontroller, compared to other functions such as trapezoidal or Gaussian functions. Computational load is critically important in real-time control applications, which is the focus of this study. The position error and position error change membership functions presented in Figure 4 use seven symbolic expressions: NB (Negative Big), NS (Negative Small), NM (Negative Medium), Z (Zero), PS (Positive Small), PM (Positive Medium), and PB (Positive Big). The membership functions developed for the control output are shown in Figure 5. All membership function parameters were determined through iterative testing and empirical validation to ensure an optimal system response.

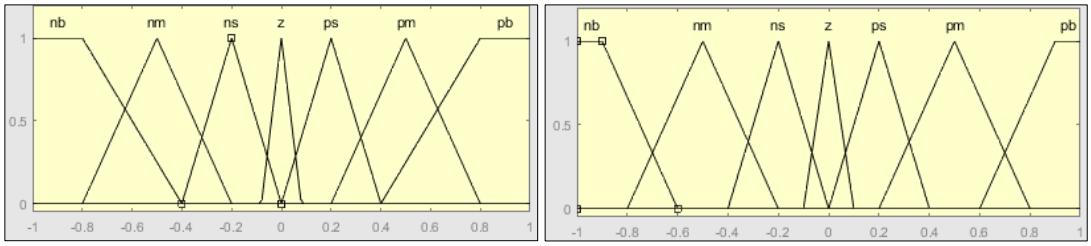

Figure 4. a) Input-1 membership functions b) Input-2 membership functions

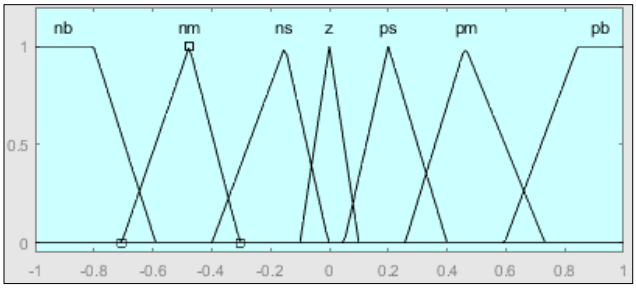

Figure 5. Output membership functions

A rule table is required to process the fuzzified error and error rate data and obtain the appropriate control output from the output membership function shown in Figure 5. The rule table in Table 2 was designed based on a heuristic approach, leveraging control engineering principles and expert knowledge. The rules were formulated by considering the expected motor response under various operating

conditions. Specifically, for large errors, stronger corrective actions are applied, while for smaller errors, finer adjustments are made to maintain stability.

Table 2. FLC Rules

	de						
e	**NB**	**NM**	**NS**	**AZ**	**PS**	**PM**	**PB**
NB	NB	NB	NB	NM	NM	NS	AZ
NM	NB	NB	NB	NM	NS	AZ	PS
NS	NB	NM	NM	NS	AZ	PS	PM
AZ	NB	NS	NS	AZ	PS	PM	PB
PS	NM	AZ	AZ	PS	PM	PB	PB
PM	NS	PS	PS	PM	PB	PB	PB
PB	AZ	PM	PM	PM	PB	PB	PB

In the constructed rule table, the "*u*" parameter represents the output value, the "*e*" parameter represents the error, and the "Δ*e*" parameter represents the change in error. As shown in Table 2, 49 different rules have been defined (Kizir et al., 2019). The range of the inputs and output membership functions is shown in Table 3.

Table 3. Range of the inputs and output membership functions

Δe / e	**INPUT-1**	**INPUT-2**	**OUTPUT**
NB	[-1 -1 -0.8 -0.4]	[-1 -1 -0.9 -0.6]	[-1.6 -1.07 -0.8016 -0.59]
NM	[-0.8 -0.5 -0.2]	[-0.8 -0.5 -0.2]	[-0.7063 -0.476 -0.304]
NS	[-0.4 -0.2 0]	[-0.4 -0.2 0]	[-0.399 -0.151 -0.00265]
Z	[-0.08 0 0.08]	[-0.1 0 0.1]	[-0.1 0 0.1]
PS	[0 0.2 0.4]	[0 0.2 0.4]	[0.05 0.2 0.4]
PM	[0.2 0.5 0.8]	[0.2 0.5 0.8]	[0.257 0.461 0.733]
PB	[0.4 0.8 1 1]	[0.6 0.9 1 1]	[0.595 0.8439 1.07 1.6]

"Input1" and "input2" represent error and error change in the motor control application, respectively, while the "output1" axis denotes the control signal generated by the FLC. The color transitions illustrate the control output values corresponding to various input combinations, revealing how the controller responds to varying conditions and enhances system performance, as shown in Figure 6. The rules and membership functions were fine-tuned through experimental validation to achieve the desired control performance.

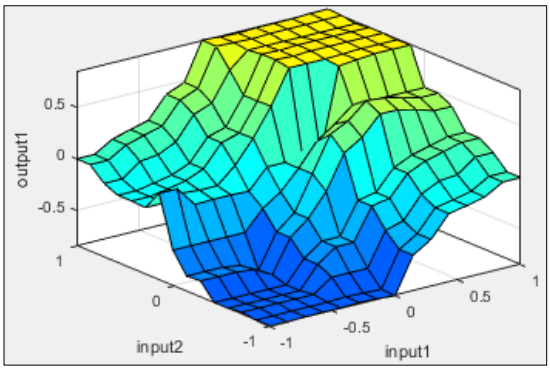

Figure 6. FLC surface

The Role of Derivative in Controller

In control systems, particularly those utilizing FLC, the derivative of the error plays a crucial role in improving the system's response to dynamic changes. The error signal, which represents the difference between the desired and actual outputs, often changes over time as the system reacts to various inputs. Taking the derivative of this error allows the controller to account for the rate of change of the error, providing valuable information about how quickly the system is approaching its desired state. By incorporating the derivative of the error, the controller becomes more responsive to rapid changes,

helping to anticipate and mitigate overshoots or oscillations. In fuzzy logic systems, this derivative term typically acts as a corrective factor, adjusting the control signal based on how fast the error is changing. However, while the derivative term enhances the controller's ability to respond to sudden variations, it may also amplify high-frequency noise, which can introduce instability if not properly filtered. Therefore, it is essential to carefully consider the impact of the derivative term in the fuzzy logic rule set, balancing responsiveness with noise immunity to achieve optimal system performance.

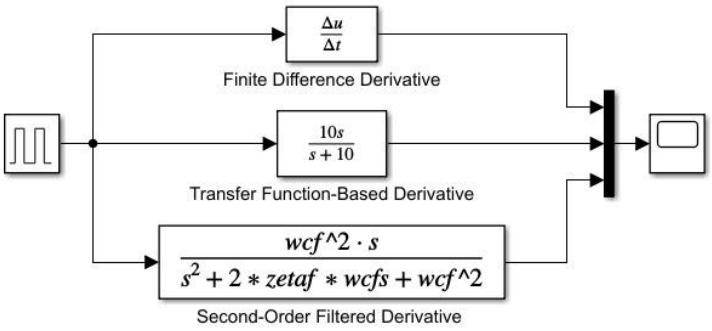

Figure 7. Observing the effect of derivative filtering in the Simulink environment

An application has been developed, as seen in Figure 7, to analyze how different derivative approaches respond to sudden changes. The approaches considered in the application are the finite difference derivative, the transfer function-based derivative, and the second-order filtered derivative.

The finite difference (FD) method, represented as *Δu/Δt*, is a simple yet effective numerical approach for approximating the derivative of a signal. This method calculates the rate of change by evaluating differences over a small-time interval, making it computationally efficient and easy to implement. However, its significant drawback is its high sensitivity to noise. In real-world scenarios, even minor fluctuations in the input signal can cause large, undesirable oscillations in the derivative output. This limitation is evident from the pronounced spikes in the response curve, demonstrating the method's instability and unsuitability for systems operating in noisy environments.

To address the noise sensitivity of numerical differentiation, a transfer function-based derivative (TFD) filter, such as Equation 8, can be employed. This method incorporates a low-pass filter characteristic, which helps attenuate high-frequency noise, resulting in a smoother derivative signal. The key advantage of this approach lies in its ability to improve the system's robustness and stability. However, the inclusion of the low-pass filter introduces a trade-off: a slight delay in the derivative response. This delay can reduce the control system's ability to react rapidly to changes, highlighting the balance that must be achieved between noise reduction and responsiveness.

A more sophisticated method involves the use of a second-order filtered derivative (SOFD), represented by the transfer function in Equation 8. This approach offers enhanced noise filtering capabilities by shaping the frequency response more precisely. The second-order filter is particularly advantageous in applications where noise suppression is critical, as it provides a smoother output while maintaining a reasonable level of responsiveness. Despite its benefits, this method requires careful parameter tuning. The bandwidth (ω_c), and damping ratio (ζ) must be appropriately selected to ensure optimal performance, as incorrect tuning can lead to instability or degraded system response. Bandwidth is expressed as $2\pi f_c$, depending on the cutoff frequency.

$$TFD = \frac{10s}{s+10},\ SOFD = \frac{{\omega_c}^2 s}{s^2 + 2\zeta\omega_c s + {\omega_c}^2} \tag{8}$$

The results of the derivative approaches against a pulse generator are shown in the graph in Figure 8. The bandwidth was chosen as 2π×5 rad/s, and the damping ratio was set to 0.9.

The FD shows high-amplitude sudden spikes when there is a significant change in the input signal. This behavior indicates that the FD method is excessively sensitive to even small fluctuations, which may be caused by noise or abrupt signal transitions. The large spikes undermine the reliability of this method in practical applications, as it amplifies high-frequency noise and causes the response to become unstable. This can lead to erratic and inefficient performance, especially in control systems with noise, making this approach unsuitable for systems with noise.

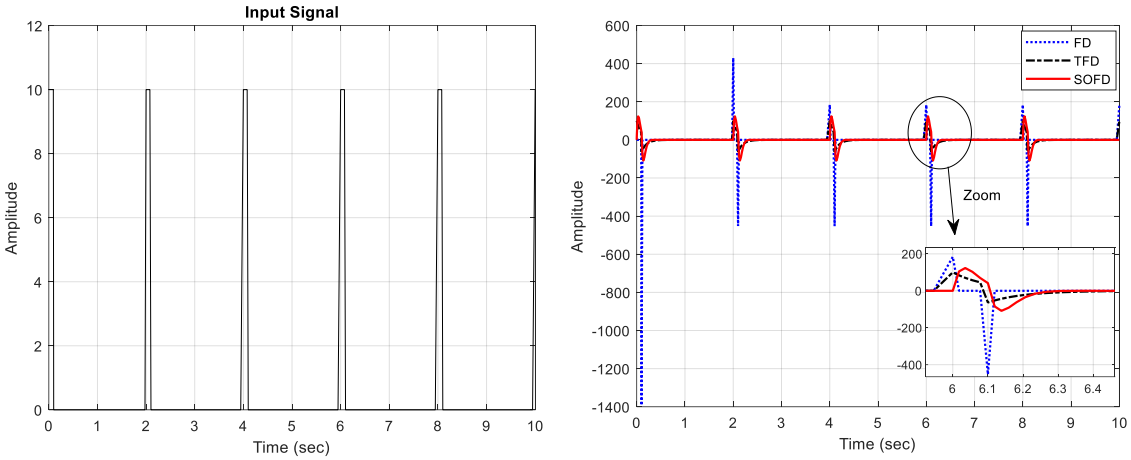

Figure 8. a) Input signal, b) derivative approach responses

The TFD is much smoother than the FD, significantly reducing oscillations. However, a slight delay in response to signal changes is observed. The transfer function smooths the derivative output with a low-pass filtering effect, which results in a slight delay in the system's response to sudden changes. This method offers a balanced approach for reducing and smoothing noise, but it comes with a slower response. While it provides an advantage in systems that require a smoother signal, the delay may be a concern in applications that demand rapid response.

The SOFD is the smoothest of all, showing less oscillation and a more stable response. Like the TFD, there is some delay, but it provides better noise suppression. The second-order filter offers a more controlled response while smoothing the derivative and effectively addressing noise, thus avoiding the excessive oscillations seen with the FD method. This approach is ideal for applications where stability and noise immunity are prioritized. Although it still introduces some delay, it is more reliable and robust in terms of overall performance.

The FD is highly sensitive to noise and unstable, making it inefficient in practical applications with noisy signals. The TFD reduces noise but introduces delay, providing a balanced approach for general use. The SOFD offers the best noise suppression and stability, making it the preferred method for systems where smooth and reliable performance is crucial. Figure 8 demonstrates the superior performance of the SOFD method, particularly in terms of durability and noise management, making it the recommended option for most control system applications. In the FLC implementation, the performance of the controller has been analyzed in detail by incorporating the TFD and SOFD techniques, respectively.

RESULTS AND DISCUSSION

Test Bench

The test setup designed for the implementation of the controllers is shown in Figure 9. In this setup, a DC motor with a gearbox and encoder, whose parameters are given in Table 1, is used. The encoder used for the position measurement of the motor has a CPR value of 168. The LMD18200 is

chosen as the motor driver. The reference motor position value is provided to the system, and the system's response is transferred to the computer via a USB-TTL converter.

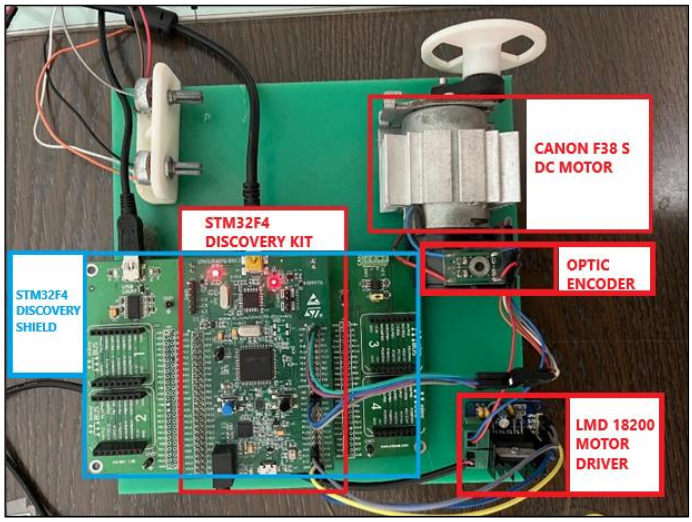

Figure 9. DC motor control test bench

The STM32F4 development kit used in the test setup is an advanced microcontroller based on the ARM Cortex M4. It is highly suitable for such projects due to its rich peripheral units, such as UART, ADC, DAC, PWM, I^2C, and SPI, and its high processing capacity. The Waijung library was used for creating the control model and programming the development board. This library can be added externally to MATLAB Simulink and used in an integrated manner with other Simulink libraries. The Waijung library is compatible with many STM32F4 microcontrollers and allows users to perform operations through Simulink blocks without writing code. This feature enables easy programming of supported STM32F4 microcontrollers and rapid prototyping. With its specified advantages, the Waijung and STM32F4 combination provides significant convenience for such projects.

Real-Time Implementation and Performance: TFD-FLC & SOFD-FLC

In this section, we present the real-time implementations of the Fuzzy Logic Controller with Transfer Function Derivative (TFD-FLC) and Fuzzy Logic Controller with Second-Order Filtered Derivative (SOFD-FLC) and evaluate their performance in practical DC motor position control applications. The Simulink model, created using the Waijung block sets and loaded onto the STM32F4 microcontroller, is shown in Figure 10. As seen from the figure, the reference signal is sent by the host application and received through the UART RX block. A summing function is applied to the reference signal received over the serial communication, and the error signal is obtained by comparing it with the motor output. This error signal is multiplied by a gain of 1/1500 in this application. The change in the error is derived using the observed transfer function and multiplied by a gain of 1/20000. This is because the membership functions designed for the error and its change are chosen to operate within the ±1 range. Multiplying the error by 1/1500 ensures that the membership functions, designed in the ±1 range, will operate within the ±1500 degree range. The region assigned to the membership functions can also be directly adjusted based on the system’s operating range.

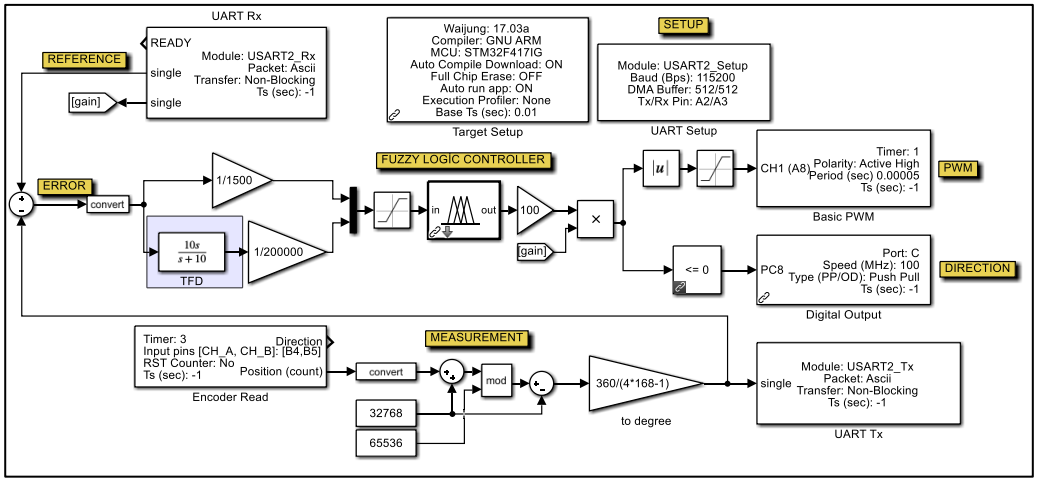

Figure 10. The control software developed in Matlab/Simulink

Instead of using FD from the Simulink blocks, the derivative is taken using the TFD. (In the SOFD-FLC Simulink model, the only modification involves updating the derivative block to use the SOFD method.) Since there are discontinuities in the error signal, high-amplitude signals are obtained when the derivative is taken.

The reference signal is received through the UART Rx block, which is configured in non-blocking mode for efficient data transmission. The motor's measured position is obtained via the *Encoder Read* block, where raw count values are processed and converted into degrees. The error signal is then computed by subtracting the measured position from the reference input.

The computed error and its rate of change are applied to the 'Fuzzy Logic Controller' block in the Simulink library. For this, both signals must be combined using a 'MUX' block. The purpose of the 'Saturation' block is to ensure that the error and its change signals remain within the designed fuzzy logic operating range. Similarly, since the output membership functions of the designed fuzzy structure are also designed in the ±1 range, the output is multiplied by a gain of 100, transforming it into a ±100 duty cycle before being applied to the PWM generation block.

The PWM output is generated using Timer 1 (Channel A8), configured as an active high signal with a period of 0.00005 seconds. This PWM signal is then sent to the motor driver to control position. The direction control is managed separately through a digital output block assigned to Port C8, which determines the motor's rotation direction based on whether the control signal is positive or negative. The controller's sampling time is set to 0.01 seconds. The UART sampling time is 0.001 seconds, operating at a baud rate of 115200. The system output is also monitored by sending it to the host application through the UART TX block. The host application model is shown in Figure 11.

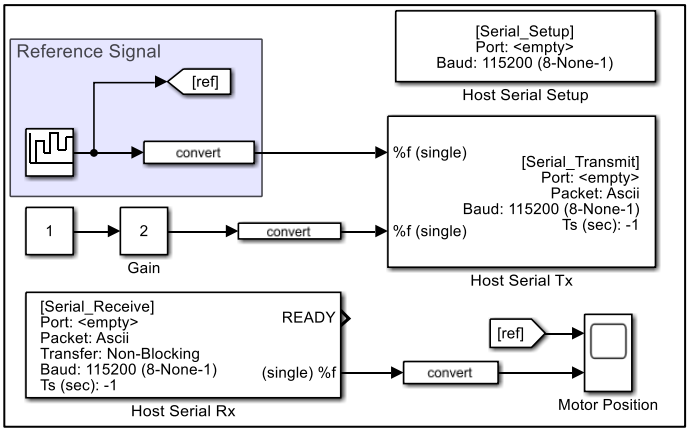

Figure 11. Host model in Matlab/Simulink

In the real-time FLC implementations, a repeating sequence stair signal was applied to the DC motor position as reference input. The use of a sequence stair rather than a single-step input introduces more challenging conditions for the controller, as it requires the system to handle multiple transitions between different reference values. This approach tests the controller's ability to maintain stability and accuracy under varying input conditions, simulating real-world scenarios where the desired position may change dynamically. By employing a sequence of steps, the system's response to sudden shifts in reference is evaluated, allowing for a more comprehensive assessment of the FLC's performance, including its robustness and adaptability to rapid changes in setpoint.

The Simulink host model transmits the reference position data. As shown in Figure 12, the real-time control outcomes for FLC using various derivative methods in response to a repeating sequence stair input can be seen. The applied reference input is defined as [0, 90, 35, 120, 50, 100] degree, with a sampling time of 1 second.

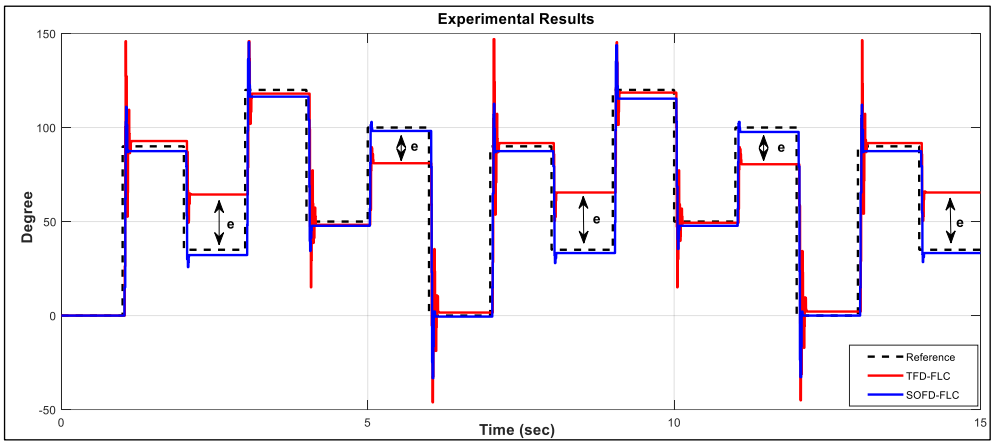

Figure 12. Experimental motor position results of TFD-FLC & SOFD-FLC

The experimental results depicted in the graph provide a comparative analysis of two control strategies, TFD-FLC and SOFD-FLC, alongside a reference signal. The reference signal, illustrated by the dashed black line, serves as the desired trajectory that both control systems aim to track. It establishes an ideal performance benchmark against which the effectiveness of each controller can be evaluated.

The TFD-FLC, represented by the red line, demonstrates a mixed performance. While this controller is capable of responding quickly to changes in the reference signal, the results indicate significant overshoot and oscillatory behavior in some regions. These oscillations suggest that while TFD-FLC provides a rapid response, it compromises on system stability and precision. The presence of overshoot indicates that the system may be prone to instability or excessive energy consumption, especially in applications requiring high accuracy.

In contrast, the SOFD-FLC, illustrated by the blue line, exhibits a smoother response that more closely follows the reference signal. This indicates an improvement in system stability and reduced error margins. The SOFD-FLC's behavior is characterized by a lower overshoot and minimized oscillations, highlighting the controller's ability to balance responsiveness with precision. The smoother performance profile suggests that the SOFD-FLC is better suited for applications where stability and minimal error are critical.

Error analysis, denoted as e on the graph, provides a quantitative assessment of each controller's deviation from the reference signal. The TFD-FLC exhibits larger error values, particularly in regions where overshoot occurs, emphasizing the limitations of this control strategy in achieving accurate tracking. Conversely, the SOFD-FLC consistently maintains lower error values, demonstrating its enhanced capability to minimize discrepancies between the actual and desired outputs. This

improvement underscores the SOFD-FLC's efficacy in achieving more reliable and controlled performance.

In particular, the results of the controller were compared and evaluated in terms of computational cost. The cost functions are defined by the root mean squared error (RMSE), mean squared error (MSE), and mean absolute error (MAE), as shown in Equations 9.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(e(t))^2}, MSE = \frac{1}{N}\sum_{i=1}^{N}(e(t))^2, MAE = \frac{1}{N}\sum_{i=1}^{N}|e(t)| \quad (9)$$

To further quantify the performance of the two control strategies, we compared the results using standard error metrics: Root Mean Squared Error (RMSE), Mean Squared Error (MSE), and Mean Absolute Error (MAE). These metrics provide a comprehensive evaluation of the controllers' effectiveness in minimizing deviations from the reference signal. The comparison results are presented in Table 4.

The SOFD-FLC achieved lower RMSE, MSE, and MAE values, demonstrating its superior ability to minimize large deviations from the reference trajectory. Specifically, the RMSE of SOFD-FLC improved by 28.2% compared to TFD-FLC, while the MSE and MAE improvements were 48.3% and 59.1%, respectively. These improvements indicate that the SOFD-FLC controller provides smoother control with fewer oscillations and lower computational cost, which is particularly advantageous for high-precision applications such as robotic manipulators and motion control systems. Furthermore, the significant reduction in MAE suggests that SOFD-FLC ensures more accurate trajectory tracking, making it more suitable for tasks where minimizing steady-state errors is critical.

Table 4. The controller performance comparison

	RMSE	MSE	MAE
TFD-FLC	0,1729	448,376	12,8692
SOFD-FLC	0,1242	231,686	5,2588

Figure 13 shows the motion of the motor shaft when the first element of the input sequence is applied. A comparison of these data in terms of performance characteristics, such as percentage overshoot, rise time, settling time, and steady-state error, is presented in Table 5.

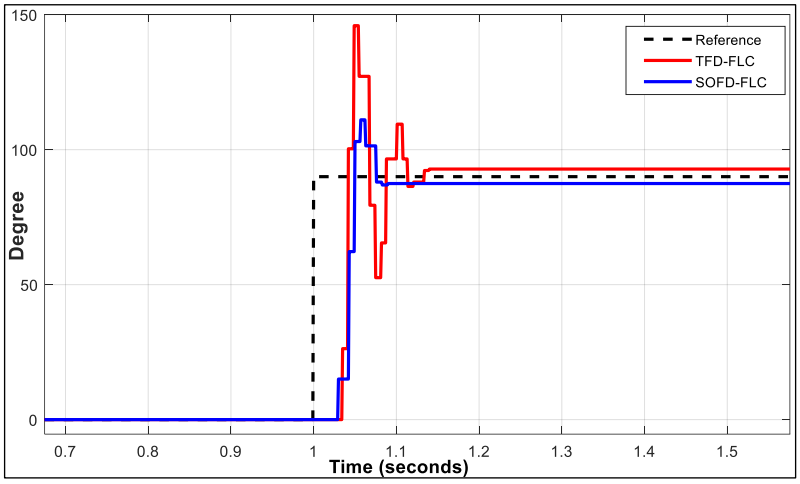

Figure 13. The first element of the input sequence – performance characteristics

The comparison between TFD-FLC and SOFD-FLC reveals that SOFD-FLC significantly outperforms TFD-FLC in key performance metrics.

While both controllers exhibit identical rise times, SOFD-FLC demonstrates a shorter settling time, lower steady-state error, and notably reduced overshoot. These improvements indicate that SOFD-FLC enhances the system's ability to reach stability faster while reducing oscillatory behavior, which

holds significant practical implications. A lower settling time ensures faster system stabilization, which is crucial in high-speed automation and robotic applications. Similarly, reduced steady-state error enhances precision in trajectory tracking, making SOFD-FLC particularly suitable for tasks requiring high accuracy, such as CNC machining and autonomous navigation. Furthermore, the lower overshoot observed in SOFD-FLC minimizes excessive oscillations, which is highly beneficial for sensitive control applications, including medical devices and stabilization systems. These findings emphasize that SOFD-FLC provides a more stable and efficient control strategy for real-world applications requiring precision and reliability.

Table 5. Performance criteria

	TFD-FLC	SOFD-FLC
Rise Time (s)	0.43	0.43
Settling Time (s)	0.135	0.083
Steady-State Error (°)	2.8167	2.5484
Overshoot (%)	50.4	23.2

CONCLUSION

This study presented a comprehensive evaluation of derivative approaches in real-time DC motor position control using FLC. By implementing and comparing the TFD and SOFD techniques, the research highlights the critical role of derivative filtering in achieving stable and noise-resistant control. The TFD method demonstrated a balanced performance, providing acceptable noise reduction and responsiveness; however, it introduced a delay that may be suboptimal for rapid system responses. Conversely, the SOFD approach exhibited superior noise suppression and overall stability, making it more suitable for environments with significant disturbances or high-frequency noise.

The mathematical modeling of the DC motor system was developed using both white-box and black-box system identification approaches. The white-box model employed the fundamental physical principles governing the DC motor dynamics, including the motor's electrical and mechanical equations, while the black-box model utilized empirical input-output data to generate a model that accurately predicts the motor's behavior. The white-box model facilitated an in-depth understanding of the system, while the black-box model provided a more flexible and data-driven alternative for system identification. Together, these models were used to evaluate and optimize the performance of the FLC-based control system under various operating conditions.

Although this study was conducted on a specific DC motor, the proposed FLC structure remains adaptable to different motor types and load conditions, provided that the membership functions and scaling factors are appropriately tuned. The model-free nature of FLC enables its generalization beyond the tested setup, making it applicable to a broader range of control systems.

These findings emphasize the importance of selecting appropriate derivative methods based on application requirements and suggest that incorporating advanced filtering techniques can significantly enhance the performance of FLC-based control systems. Future work could explore adaptive tuning mechanisms for further optimizing the performance under varying operational conditions.

In the blind article, this field should be left blank. This section should be written on the title page.

Conflict of Interest

The article authors declare that there is no conflict of interest between them.

Author’s Contributions

The authors declare that they have contributed equally to the article.

REFERENCES

Adukwu, O., Dehunsi, O. A., & Adeyeri, K. (2023). Modelling of Direct Current (DC) Motor for Performance Improvement using Model Parameter Estimation. Journal Of Engineering Research Innovation And Scientific Development, 1(2), 36–41. https://doi.org/10.61448/JERISD12235

Akbari-Hasanjani, R., Javadi, S., & Sabbaghi-Nadooshan, R. (2014). DC motor speed control by self-tuning fuzzy PID algorithm. Http://Dx.Doi.Org/10.1177/0142331214535619, 37(2), 164–176. https://doi.org/10.1177/0142331214535619

Al-Bargothi, S. N., Qaryouti, G. M., & Jaber, Q. M. (2019). Speed control of DC motor using conventional and adaptive PID controllers. Indonesian Journal of Electrical Engineering and Computer Science, 16(3), 1221–1228. https://doi.org/10.11591/ijeecs.v16.i3.pp1221-1228

Asadi, F. (2018). Comparison Of Different DC Motor Modeling Techniques. Journal of Electronic Research and Application, 2(2). https://doi.org/10.26689/JERA.V2I2.333

Frances, A., Asensi, R., & Uceda, J. (2019). Blackbox Polytopic Model with Dynamic Weighting Functions for DC-DC Converters. IEEE Access, 7, 160263–160273. https://doi.org/10.1109/ACCESS.2019.2950983

Gebremariam, S. F., & Alemu, B. S. (2023). A Review on DC Motor Drive Controlling Schemes, Optimization Techniques and Future Trends. International Journal of Research Publication and Reviews, 4(12), 3507–3514. https://doi.org/10.55248/GENGPI.4.1223.0127

Gu, D., Zhang, J., & Gu, J. (2015). Brushless DC motor speed control based on predictive functional control. Proceedings of the 2015 27th Chinese Control and Decision Conference, CCDC 2015, 3456–3458. https://doi.org/10.1109/CCDC.2015.7162520

Hidayati, Q., & Prasetyo, M. E. (2016). Pengaturan Kecepatan Motor DC dengan Menggunakan Mikrokontroler Berbasis Fuzzy-PID. JTT (Jurnal Teknologi Terpadu), 4(1). https://doi.org/10.32487/JTT.V4I1.123

Kizir, S., Kelekci, E., & Yaren, T. (2019). Matlab Simulink Destekli Gerçek Zamanlı Kontrol Teori ve Mühendislik Uygulamaları.

Kroičs, K., & Būmanis, A. (2024). BLDC Motor Speed Control with Digital Adaptive PID-Fuzzy Controller and Reduced Harmonic Content. Energies, 17(6), 1311. https://doi.org/10.3390/en17061311

Liu, S., Zhang, H., & Pang, H. (2025). Finite-time adaptive fuzzy constrained control of uncertain switched nonlinear systems with zero dynamics. Journal of the Franklin Institute, 362(3), 107513. https://doi.org/10.1016/J.JFRANKLIN.2025.107513

Ma'arif, A., & Çakan, A. (2021). Simulation and Arduino Hardware Implementation of DC Motor Control Using Sliding Mode Controller. Journal of Robotics and Control (JRC), 2(6), 582–587. https://doi.org/10.18196/JRC.26140

Moaveni, B., Masoumi, Z., & Rahmani, P. (2023). Introducing Improved Iterated Extended Kalman Filter (IIEKF) to Estimate the Rotor Rotational Speed, Rotor and Stator Resistances of Induction Motors. IEEE Access, 11, 17584–17593. https://doi.org/10.1109/ACCESS.2023.3244830

Naziris, A., Frances, A., Asensi, R., & Uceda, J. (2020). Black-box small-signal structure for single-phase and three-phase electric vehicle battery chargers. IEEE Access, 8, 170496–170506. https://doi.org/10.1109/ACCESS.2020.3024534

Ortatepe, Z. (2023). Genetic Algorithm based PID Tuning Software Design and Implementation for a DC Motor Control System. Gazi University Journal of Science Part A: Engineering and Innovation, 10(3), 286–300. https://doi.org/10.54287/GUJSA.1342905

Prasad, L. B., Tyagi, B., & Gupta, H. O. (2014). Optimal control of nonlinear inverted pendulum system using PID controller and LQR: Performance analysis without and with disturbance input. International Journal of Automation and Computing, 11(6), 661–670. https://doi.org/10.1007/S11633-014-0818-1/TABLES/4

Raza, W., Adzikya, D., Mehmood, S., Wasti, S. R., Hussain, M. J., Ahmad, A., … Raza, S. (2024). Fuzzy Logic Speed Regulator for D.C. Motor Tuning. JTAM (Jurnal Teori Dan Aplikasi Matematika), 8(1), 36–49. https://doi.org/10.31764/jtam.v8i1.16919

Sadi, S. (2020). DC Motor Speed Control Using Mamdani Fuzzy Logic Based on Microcontroller. Jurnal Teknik, 9(2). https://doi.org/10.31000/JT.V9I2.3676

Saleem, O., & Omer, U. (2017). EKF-based self-regulation of an adaptive nonlinear PI speed controller for a DC motor. Turkish Journal of Electrical Engineering and Computer Sciences, 25(5), 4131–4141. https://doi.org/10.3906/elk-1611-311

Sami, S. S., Obaid, Z. A., Muhssin, M. T., & Hussain, A. N. (2021). Detailed modelling and simulation of different DC motor types for research and educational purposes. International Journal of Power Electronics and Drive Systems (IJPEDS), 12(2), 703–714. https://doi.org/10.11591/IJPEDS.V12.I2.PP703-714

Shah, J., Okasha, M., & Faris, W. (2018). Gain scheduled integral linear quadratic control for quadcopter. International Journal of Engineering & Technology, 7(4.13), 81–85. https://doi.org/10.14419/IJET.V7I4.13.21334

Valenzuela, F. A., Ramírez, R., Martínez, F., Morfín, O. A., & Castañeda, C. E. (2020). Super-Twisting Algorithm Applied to Velocity Control of DC Motor without Mechanical Sensors Dependence. Energies 2020, Vol. 13, Page 6041, 13(22), 6041. https://doi.org/10.3390/EN13226041

Xue, C., Zhu, H., & Yu, B. (2012). Modeling and Simulation of Parameter Self-Tuning Fuzzy PID Controller for DC Motor Speed Control System. Applied Mechanics and Materials, 195–196, 1003–1007. https://doi.org/10.4028/WWW.SCIENTIFIC.NET/AMM.195-196.1003