IMPROVEMENT AND CONTROL OF THE SPEED RESPONSE OF THE PERMANENT MAGNET SYNCHRONOUS MOTOR DRIVE USING A FUZZY – PI CONTROLLER

T. N. Elgargani [1,*]
A. A. Hudoud [2,*]
S. O. Abid [3]

Received: 16/02/2025
Revised: 25/03/2025
Accepted: 26/03/2025

1 Department of Electrical and Computer Engineering, University of Elmergib, Faculty of Engineering, Alkhums, Libya,
Email: tngargany@elmergib.edu.ly
2 Computer and systems engineering Department, University of Azzytouna, Faculty of Engineering, Tarhuna, Libya.
3 Department of Electrical and Electronices Engineering, University of Alasmarya Islamic, Faculty of Engineering, Zliten, Libya,
Email: s.abied@asmarya.edu.ly
*Corresponding Author's Email: a.hdoud@azu.edu.ly

Improvement And Control of The Speed Response of The Permanent Magnet Synchronous Motor Drive Using a Fuzzy – PI Controller

T. N. Elgargani
Department of Electrical and Computer Engineering, University of Elmergib, Faculty of Engineering, Alkhums, Libya
tngargany@elmergib.edu.ly

A. A. Hudoud
Computer and systems engineering Department, University of Azzytouna, Faculty of Engineering, Tarhuna, Libya,
a.hdoud@azu.edu.ly

S.O. Abid
Department of Electrical and Electronices Engineering, University of Alasmarya Islamic, Faculty of Engineering, Zliten, Libya, Email:
s.abied@asmarya.edu.ly

***Abstract*—**High-speed and high-performance electric motors are designed to reach a high level of demand control. The permanent magnet synchronous motors (PMSMs) drive has a non-linear model that is not easy to deal with using traditional control methods when controlling the three phase motors because of their nature, (intricate highly non-linear model). Therefore, neural networks controllers compared with fuzzy logic controllers (FLCs) are getting more attention among researchers, as they can be used for such systems. The neural networks controller relies on training of this mathematical model, and the fuzzy controller also relies on experience. The performance of these two controllers were compared to each other in terms of output response. As all the real systems exhibit non-linear behavior, conventional PI (Proportional-Integral) controllers are unable to provide good and acceptable results. For this reason, when designing intelligent control systems, the corresponding model for simulation should reflect all characteristics of the real system to be controlled. The basic idea of this paper is to apply the fuzzy-PI controller on PMSMs drive and compare the obtained results with the traditional PI. Also, one intelligent controller, which is the NN (Neural Network) controller, is applied and its performance is simulated and studied. MATLAB/SIMULINK environment is used for design, implementation and testing. Therefore, the speed and torque of the PMSMs drive can be controlled satisfactorily. Finally, simulation results have shown decent results in the improvement of the system behavior.

Keywords— Fuzzy logic control, Neural Network controller, Permanent Magnet Synchronous Motors (PMSMs) drive, PI controller, Speed control, Torque.

I. INTRODUCTION

The escalating cost of fuel has intensified the focus on Permanent Magnet Synchronous Motor (PMSM) systems, driven by the imperative for enhanced energy efficiency. PMSMs are recognized for their high efficiency, reduced maintenance costs, and superior performance, rendering them critical components in energy conservation strategies. In the context of PMSM speed regulation, scalar control offers a simplified solution for applications where precise speed control is not paramount. Conversely, Field Oriented Control (FOC) and Direct Torque Control (DTC) are prevalent in industrial applications, valued for their high efficiency, operational simplicity, structural robustness, and reliability. These advanced control methodologies facilitate improved performance and efficiency. PMSM motors, characterized by their permanent magnet flux, exhibit exceptional efficiency by minimizing heat dissipation; however, operational limitations constrain their maximum achievable speed [1]. Surface-mounted PMSMs, favored for their compact design and high efficiency, find extensive application in wind power generation and industrial transmission systems. Achieving high-performance control typically necessitates the integration of costly speed and position sensors. To address this, sensorless control techniques have been developed. Current sensorless methods can be broadly categorized into high-frequency signal injection and motor parameter-based estimation. However, challenges such as chattering in commonly employed algorithms necessitate the investigation of alternative control strategies, with a critical consideration of the trade-offs between noise reduction and system phase delays, as detailed in [2].

In precision-centric applications such as robotics and mechanical processes, Permanent Magnet Synchronous Motors (PMSMs) play an indispensable role [3]. However, their performance is often compromised by uncertainties including noise, external loads, and friction forces. To mitigate these challenges, advanced control techniques, such as fuzzy logic and neural networks, have been employed to achieve fine regulation of motor speed and position. Field-Programmable Gate Arrays (FPGAs) facilitate the implementation of these algorithms by offering programmable hard-wired features, rapid computational speeds, and low power consumption. This integration of PMSMs, intelligent controls, and FPGA technology aims to enhance multi-axis robotic precision in machining and assembly.

The widespread adoption of PMSMs in machine tools is attributed to their streamlined design, broad speed range, and operational efficiency [4]. Despite the availability of real-time speed and position signal detection through encoders, complex machining environments present challenges, particularly in the form of encoder failures. Recent advancements in sensorless control, while promising, lack high-performance algorithms necessary for robust operation.

Consequently, research efforts are focused on developing advanced sensorless control strategies for high-speed PMSM systems, addressing limitations associated with existing methods such as back electromotive force and sliding mode observers, with an emphasis on improved stability and robustness.

The ubiquity of AC motors in both household and industrial settings underscores their importance [5]. While asynchronous motors (AMs) remain prevalent due to their simplicity and durability, PMSMs are gaining popularity due to their superior power density and efficiency. However, the nonlinear nature of PMSM systems poses challenges for precise regulation. This necessitates the development of high-performance controllers capable of enhancing stability and overcoming the limitations of traditional Proportional-Integral-Derivative (PID) controllers. The expanding utilization of PMSMs in industrial manufacturing highlights the need for advanced control methodologies, including PI control, adaptive control, model-referenced adaptation, Sliding Mode Control (SMC), H_∞ control, and Internal Model Control (IMC) [6]. Despite the widespread use of PI control, its parameter dependence in dynamic work environments hinders optimal performance. Similarly, issues with adaptive control laws and high-frequency jitter in conventional Sliding Mode Observers (SMOs) limit their broader industrial applicability.

The increasing prominence of Permanent Magnet Synchronous Machines (PMSMs) in industry is driven by their low inertia, high mass torque, and minimal maintenance requirements. In variable speed drives, the integration of AC machines and static converters offers new possibilities. Ensuring dynamic drive performance, including steady-state accuracy, overload capability, and disturbance resilience, is crucial. To achieve this, a decoupled control approach using vector control methods is adopted. Traditional controllers often struggle with stability due to parameter variations. To overcome this, innovative strategies employing adaptive fuzzy logic control (AFLC), a nonlinear system integrating algorithmic control laws, have been investigated. This approach, exemplified by an adaptive gain fuzzy controller, effectively mitigates external disturbances, thereby improving control precision in PMSMs with direct flux orientation driven by voltage inverters with hysteresis current control.

PMSMs are essential in various AC speed drive applications, encompassing power electronics, sensors, and high-speed microprocessors [7]. Control system-based PMSM drives, recognized for their dynamic response, low noise, and high efficiency, are pivotal in both domestic and industrial settings [8]. However, traditional speed sensing methods pose challenges related to robustness, cost, and volume. Research is therefore focused on developing motor speed-based sensorless control techniques to enhance the overall performance and reliability of PMSM drive systems. PMSMs are also characterized by their low moment of inertia, high start-up torque, and significant power in numerous industrial applications. To maximize torque in both induction motors and PMSM drives, control techniques such as Field-Oriented Control (FOC) and Direct Torque Control (DTC) with Space Vector Pulse Width Modulation (SVPWM) are essential. Intelligent control methods, particularly fuzzy logic, have demonstrated effectiveness in refining system modeling, speed control, and estimation for PMSM drives, indicating their potential for enhanced performance [9].

In response to these challenges and to further advance control strategies for PMSM drives, this research introduces a novel intelligent controller based on a fuzzy-PI architecture. The performance of this proposed controller is rigorously compared with that of a neural network controller and a conventional PI controller, providing a comprehensive evaluation of its efficacy.

II. PMSM MOTOR DRIVE SYSTEM

The PMSM drive system, presented in figure (1), is characterized by five primary functional blocks: a controller, the d-q/abc coordinate transformation, pulse width modulation (PWM), the abc/d-q coordinate transformation, and the PMSM. The controller and PMSM model, which are the focus of this analysis, will be addressed individually in the ensuing subsections.

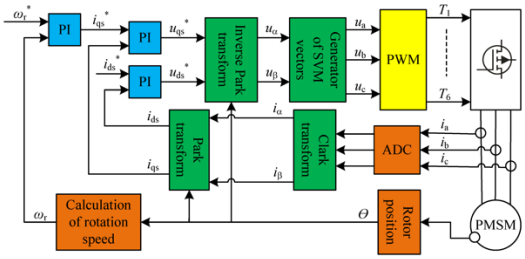


Fig. 1. PMSM drive system

A. *Control system of fuzzy logic (CSOFL)*

The proposed control methodology utilizes a fuzzy logic-based automatic controller to achieve desired system behavior through a self-regulating mechanism. The inherent advantages of fuzzy logic, namely its ability to operate using linguistic rules, render it a powerful tool for intelligent control applications. As noted by prior research [12], fuzzy control is often developed based on the experiential knowledge of process operators or design technologists. This approach contrasts with the conventional proportional-integral (PI) controller, which, while widely adopted in industrial settings due to its simplicity and ease of implementation in both digital and analog forms, lacks the adaptability of fuzzy logic.

Fuzzy controllers, however, necessitate a comprehensive understanding of fuzzy logic principles and the application of membership functions. The non-linear nature and complexity of mathematically characterizing fuzzy logic controllers (FLCs) necessitate the use of approximations, posing challenges for stability analysis. Moreover, while fuzzy controllers offer the potential for finer tuning compared to

traditional PI controllers, this increased precision also complicates error modification. Consequently, fuzzy controllers can be conceptualized as artificial decision-making systems capable of real-time implementation within closed-loop configurations. These systems process output data, y(t), compare it to a reference input, r(t), and determine the appropriate process input, u(t), to meet performance objectives. The development of a fuzzy controller relies on the acquisition of data representing the actions of the artificial decision-maker within the closed-loop system.

Data acquisition can involve leveraging the expertise of human decision-makers performing control tasks or independently developing a rule set based on a dynamic model of the system. These rules are typically formulated as "IF-THEN" statements, defining the relationship between the process output, reference input, and desired process input. Once these rules are incorporated into the rule base and an appropriate inference mechanism is selected, the system undergoes testing to validate its compliance with closed-loop control requirements. The design of a fuzzy logic control system generally encompasses three primary stages: (I) selection of the fuzzy controller's inputs and outputs; (II) determination of necessary preprocessing for controller inputs and potential post-processing for outputs; and (III) construction of the fuzzy controller's four constituent components, as illustrated in figure (2) [12].

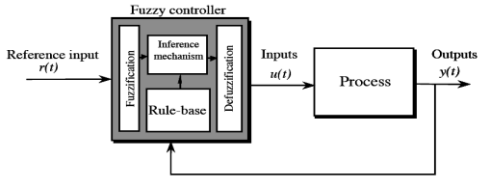


Fig. 2. Fuzzy Controller (FC) with process

B. *Fuzzy-PI Controller*

In the context of motor control applications, the implementation of a fuzzy logic controller is frequently integrated with either a proportional-integral (PI) or proportional-derivative (PD) controller. The selection of a PI controller in this study is predicated upon its demonstrated superior overall system performance compared to its PD counterpart. Moreover, the inherent limitations of PD fuzzy logic controllers, specifically their propensity to generate steady-state errors due to the absence of integral action, further justify the choice of a PI-based approach [10]. The architectural configuration of the implemented control system is presented in figure (3).

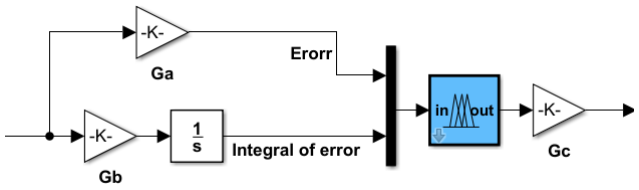


Fig. 3. Fuzzy logic PI controller

The control system incorporates several adjustable gains to optimize the output response. Specifically, G_c represents the gain applied to the output of the fuzzy logic controller, while G_a and G_b denote the proportional and integral tuning gains, respectively, associated with the error and the integral of the error. These gains serve as critical parameters for fine-tuning the system's dynamic behavior and enhancing its performance characteristics [10]. The judicious adjustment of these gains is essential for achieving the desired control objectives and mitigating potential performance degradations.

C. *Fuzzification*

Fuzzification is the process of transforming crisp numerical variables into linguistic variables, known as fuzzy numbers. The implemented fuzzy logic controller, depicted in figure (4), utilizes two inputs, the error (E) and the integral of the error (IE), to generate a single output. In the simulation, membership functions are designed with a 50% overlap, representing a logical and objective selection. Seven membership functions [Negative Big (NB), Negative Medium (NM), Negative Small (NS), Zero (ZE), Positive Small (PS), Positive Medium (PM), and Positive Big (PB)] are employed for the E, IE, and output variables, as illustrated in figure (5) (a), (b), and (c).

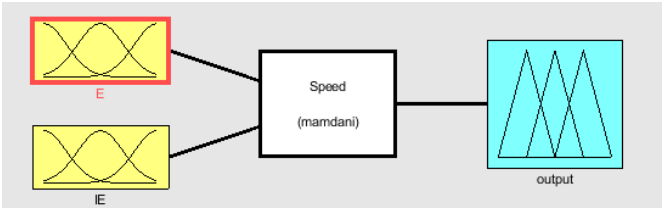
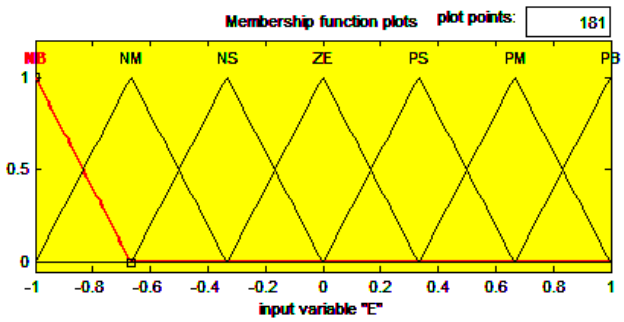
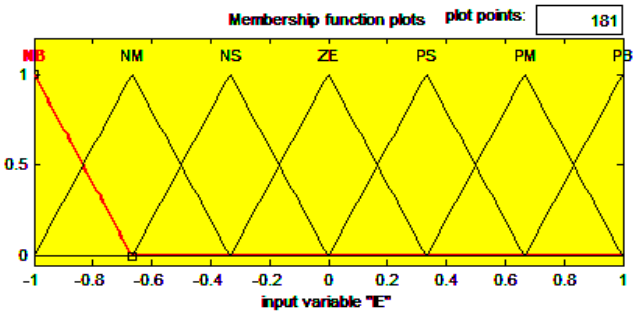


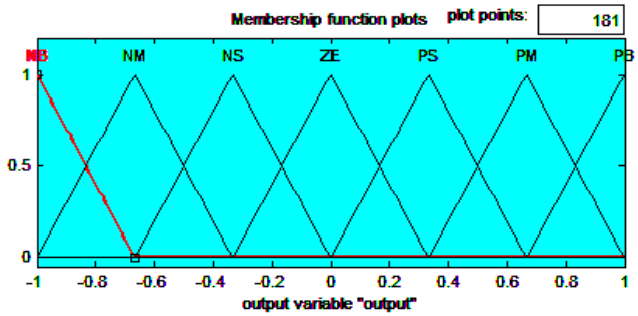
Fig. 4. FIS editor



(a)



(b)



(c)

Fig. 5. Membership functions

The scaling range for all variables is [-1, 1]. Table 1 presents the fuzzy sets defined for E, IE, and the output. For simplicity, symmetrical triangular membership functions are adopted. The linguistic terms correspond to: ZE (Zero), PS (Positive Small), PM (Positive Medium), PB (Positive Big), NB (Negative Big), NM (Negative Medium), and NS (Negative Small).

TABLE 1: Crisp to fuzzy set mapping for E, IE and output

Crisp set	Fuzzy set
-0.6666 — -1	Negative Big Error (*NB*)
-0.3334 — -1	Negative Medium Error (*NM*)
0— -0.6666	Negative Small Error (*NS*)
0.3334 — - 0.3334	Zero error (*ZE*)
0.6666 — 0	Positive Small error (*PS*)
1— 0.3334	Positive Medium error (*PM*)
1— 0.6666	Positive Big error (*PB*)

D. The rule base

When a system having seven membership functions, 49 rules will be obtained. The rule base for seven membership functions is shown in Table (2).

TABLE 2: Rule base for seven membership functions

IE	E						
	NB	***NM***	***NS***	***ZE***	***PS***	***PM***	***PB***
NB	NB	NB	NB	NB	NM	NS	ZE
NM	NB	NB	NB	NM	NS	ZE	PS
NS	NB	NB	NM	NS	ZE	PS	PM
ZE	NB	NM	NS	ZE	PS	PM	PB
PS	NM	NS	ZE	PS	PM	PB	PB
PM	NS	ZE	PS	PM	PB	PB	PB
PB	ZE	PS	PM	PB	PB	PB	PB

E. PI controller Modeling and simulation

The scope of this paper is limited to the application of a proportional-integral (PI) controller. The inclusion of integral control is motivated by its ability to minimize or eliminate steady-state errors; however, this is often associated with a trade-off in transient response performance. The mathematical description of the PI controller in the time domain is presented below, while figure (6) illustrates its implementation in the MATLAB/Simulink platform.

$$u(t) = k_p[R(t) - Y(t)] + k_i \int_0^t [R(t) - Y(t)]dt \quad (1)$$

$$u(t) = k_p e(t) + k_i \int_0^t e(t)dt \quad (2)$$

Where:
$\boldsymbol{u(t)}$ - controllers output signal
$\boldsymbol{e(t)}$ - controllers input error signal
$\boldsymbol{k_p}$ - proportional control gain
$\boldsymbol{k_i}$ - integral control gain

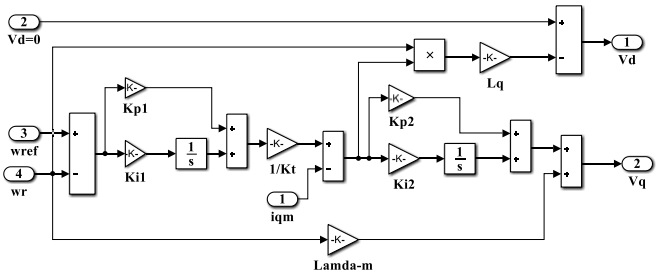


Fig.6. PI- controller MATLAB / Simulink

F. The d-q Modelling and Simulation of the PMSM drive

Vector control, also known as field-oriented control, facilitates independent control of torque and flux in permanent magnet synchronous motors (PMSMs) by transforming the three-phase stator currents into a two-axis d-q rotating reference frame [11]. This decoupling allows the PMSM to emulate the behavior of a separately excited DC machine, effectively linearizing the inherently nonlinear motor model. The application of vector control ensures a direct relationship between the control inputs and the motor's dynamic response.

To generate a rotating magnetic field and drive the rotor, balanced currents are injected into the ABC stator windings. The vector control strategy employs Park and inverse Park transformations to maintain equivalent current relationships between the synchronously rotating d-q frame and the stationary α-β and ABC stator frames. Figure (7) illustrates the PMSM vector diagram, highlighting the phase relationship between the d-axis current and the rotor flux. Specifically, the d-axis stator flux must be aligned with the

rotor motion, while the q-axis flux lags by 90 degrees. Consequently, an increase in the d-axis stator flux, when aligned with the rotor motion, enhances the rotor speed, leading to a corresponding increase in the net air gap flux [11].

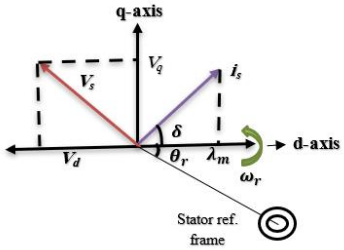


Fig. 7. PMSM's phasor representation

$$i_a = i_s \sin(\omega_r t + \delta) \quad (3)$$

$$i_b = i_s \sin\left(\omega_r t + \delta - \frac{2\pi}{3}\right) \quad (4)$$

$$i_c = i_s \sin\left(\omega_r t + \delta - \frac{2\pi}{3}\right) \quad (5)$$

Where $\theta_r = \omega_r t$, Using the phasor representation, we obtain:

$$i_q = i_s \begin{bmatrix} \sin\delta \\ \cos\delta \end{bmatrix} \quad (6)$$

If $i_d = 0$ by $\delta = 90^\circ$ Hence, the equation for electric torque:

$$T_e = \left(\frac{3}{2}\left(\frac{p}{2}\lambda_m i_q\right)\right) \quad (7)$$

One can get a constant torque, if i_q is constant. Hence, only the quadrature axis current can alter the electric torque

voltage of the d-axis stator:

$$V_d = R_i + L_d \frac{d_{i_d}}{dt} - L_q \omega_s i_q \quad (8)$$

voltage of the q-axis stator:

$$V_q = R_{i_q} + L_q \frac{d_{i_q}}{dt} - L_d \omega_s i_d + \omega_s \lambda_{af} \quad (9)$$

Magnetic flux linkage on the d axis:

$$\lambda_d = L_d \lambda_d + \lambda_{af} \quad (10)$$

Magnetic flux linkage on the q-axis:

$$\lambda_q = L_q i_q \quad (11)$$

Electromagnetic torque

$$T_e = J\frac{d}{dt}\omega_r + B\omega_r + T_l \quad (12)$$

Using the torque equation

$$T_e = K_t i_q + \frac{3}{2}\frac{P}{2}\left(L_d - L_q\right) i_d i_q \quad (13)$$

$$K_t = \frac{3}{2}\frac{P}{2}\lambda_m \quad (14)$$

Where: λ_m Permanent magnet flux, and $\omega_r = \frac{2}{p}\omega_s$

Speed of the rotor in angular frequency

$$\omega_e = \int \frac{1}{\omega_r}\left[\frac{1}{J}\frac{P}{2}\left(T_s - T_m - B\frac{2}{P}\omega_s\right)\right] \quad (15)$$

Depending on the above-mentioned equations, the motor can be simulated using the MATLAB / Simulink environment as demonstrated in figure (8).

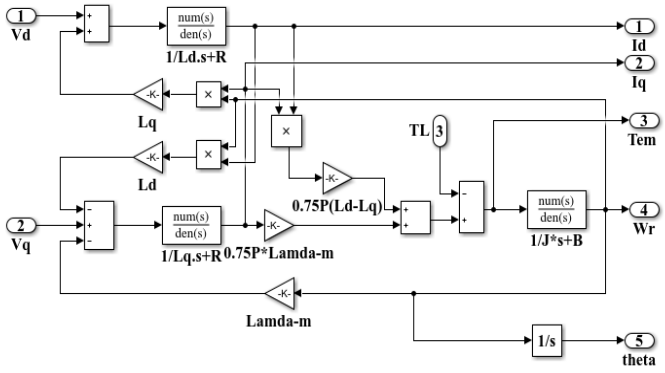


Fig.8. MATLAB / Simulink of the d-q Model of PMSM

G. Pulse Width Modulation (PWM)

Pulse width modulation (PWM), also referred to as pulse duration modulation (PDM) or pulse time modulation (PTM), represents an analog modulation technique where the pulse carrier's width, time, or length is varied proportionally to the instantaneous amplitude of the modulating message signal. This methodology maintains a constant signal amplitude while modulating the pulse width. To ensure amplitude stability, amplitude limiters are incorporated, which effectively mitigate noise by clipping the amplitude to a predetermined level.

Various forms of PWM are employed, and examples are illustrated in figures (9)(a), (b), and (c), which depict pulse width modulated waveforms within discrete time windows [13]. These figures demonstrate the principle of PWM, showcasing the relationship between the message signal's amplitude and the resulting pulse width variation.

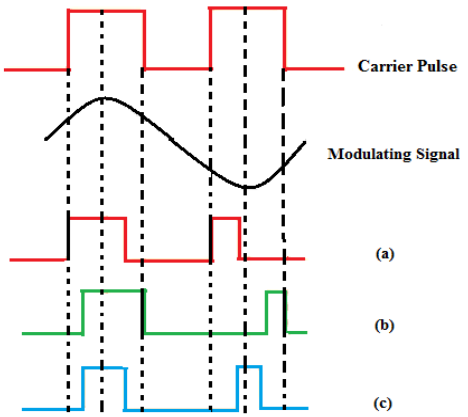


Fig.9. Waves with modulated pulse width and distinct time slots

H. Simulation of Pulse Width Modulation (PWM)

The generation of a three-phase sinusoidal pulse width modulation (PWM) waveform within the Simulink environment is facilitated by the utilization of source, repeating sequence, SUM, and SWITCH blocks. Specifically, the repeating sequence block provides the necessary triangular carrier waveform, while the source block generates the three-phase sinusoidal modulation signals. As depicted in figure (10), the Simulink implementation of the three-phase sinusoidal PWM inverter accepts the carrier frequency, modulation index, and fundamental frequency as input parameters, and produces three-phase voltage signals as its output.

For the purposes of this investigation, and to maintain model simplicity, only the PWM inverter section of the power circuit was modeled. Consequently, a constant DC link voltage, denoted as V_{dc}, is assumed. This simplification allows for a focused analysis of the PWM modulation process without the added complexity of DC link voltage variations.

$$V_{an} = \frac{(2V_{ao} - V_{bo} - V_{co})}{3} \quad (16)$$

$$V_{bn} = \frac{(2V_{bo} - V_{co} - V_{ao})}{3} \quad (17)$$

$$V_{cn} = \frac{(2V_{co} - V_{bo} - V_{ao})}{3} \quad (18)$$

Where:

A	phase- A
B	phase -B
C	phase -C
PMSM	Permanent Magnet Synchronous Motor
PWM	Pulse Width Modulation
rpm	Revolution Per minute
PI	Proportional Integral
$\boldsymbol{T_{em}}$	Electromagnetic Torque
$\boldsymbol{T_L}$	Load Torque
d	Direct Polar Axis
q	Quadrature or Interpolar Axis
$\boldsymbol{V_q}$	q-axis voltage
$\boldsymbol{V_d}$	d-axis voltage
$\boldsymbol{R_s}$	stator resistance
$\boldsymbol{i_q}$	q-axis current
$\boldsymbol{i_d}$	d-axis current
$\boldsymbol{\omega_r}$	electrical speed
$\boldsymbol{L_d}$	Direct-axis inductance
$\boldsymbol{L_q}$	Quadrature-axis inductance
$\boldsymbol{\lambda_q}$	flux linkage due q axis
$\boldsymbol{\lambda_d}$	flux linkage due d axis
$\boldsymbol{\lambda_m}$	Permanent magnet flux
λ	flux linkage

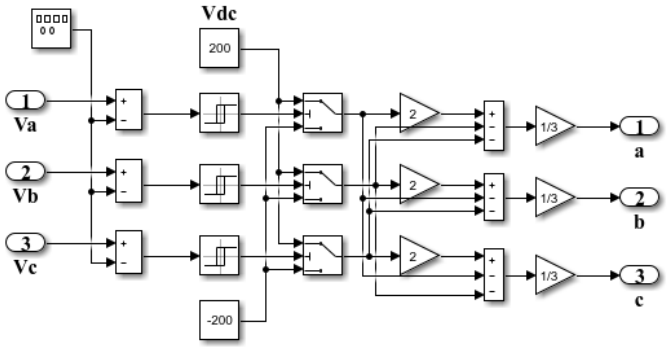


Fig.10. MATLAB / Simulink of Pulse Width Modulation

1. Simulation of the whole System:

The simulation of the complete PMSM drive system was conducted within the MATLAB/Simulink environment. To achieve a modular and hierarchical system representation, the model was constructed using two distinct block types: a single mask-able block and a lower-level block, as illustrated in figure (11). This dual-block structure facilitates organized system modeling and allows for parameter encapsulation within the mask-able blocks. The specific control strategies employed for the PMSM drive simulation encompassed both speed and current control loops are:

- PI speed controller with PWM current controller.
- Fuzzy-PI Controller with PWM current controller.
- Table (3) illustrate the PMSM drive simulation setting [14].

TABLE 3: The PMSM drive parameters

Parameters	Symbol	Value
Rated stator voltage	V_S	380 V
Load Torque	T_L	6.45 N.m
Moment of inertia	J	0.015 kg.m^2
Armature resistance	R_a	3.59 Ω
Per phase inductance	L Ph	0.0435 H
Rated Frequency	F	60 Hz
Pole pairs	*p*	6
Nominal rotor speed	nn	125.6 rad/sec (1200 rpm)
Permanent magnet flux	λ_m	0.545 Vs

Direct-axis inductance	L_d	0.036 H
Quadrature-axis inductance	L_q	0.051 H
constant	K_E	0.148 V.rad/s

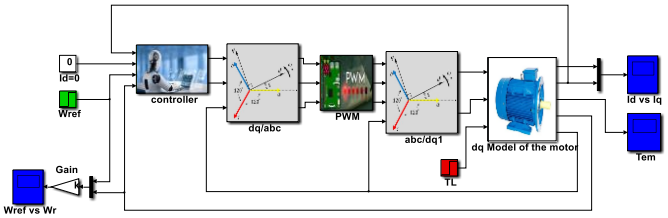


Fig.11. MATLAB / Simulink of the whole system

III. Simulation results

The efficacy of the proposed control strategy was evaluated through comparative simulations employing a standard proportional-integral (PI) controller, a fuzzy-PI based controller, and a neural network (NN) controller. To assess system reliability, a step input signal of unit amplitude was individually applied to each controller. Two distinct simulation conditions were considered: a nominal case and a case with an applied load. The resulting performance metrics were categorized and analyzed to determine the impact of each controller under varying operational scenarios.

The simulation results, depicted in the subsequent figures, provide a comprehensive overview of the system's response to the applied step input. Quantitative analysis of the system's time-domain specifications, under both nominal and loaded conditions, is presented in Tables (4) and (5). These tables facilitate a direct comparison of the controllers' performance, highlighting the advantages and disadvantages associated with each approach under different operating loads.

I. The results that are obtained by applying the proposed Fuzzy-PI controller:

- Nominal case

Under no-load conditions, the closed-loop system's transient response with unity feedback is depicted in figure (12) (a), (b), and (c), The output speed profile exhibits a smooth trajectory with an overshoot of 0.0243, indicating that the fuzzy-PI controller satisfies the performance criteria. Furthermore, the system demonstrates a settling time of 0.0572 seconds and a rise time of 0.0392 seconds. However, the steady-state error of 3.78987%.
When the setpoint was varied by 764 rpm, 955 rpm, and 1200 rpm, the overshoot, rise time, settling time, and steady-state error were as shown in Table (6).

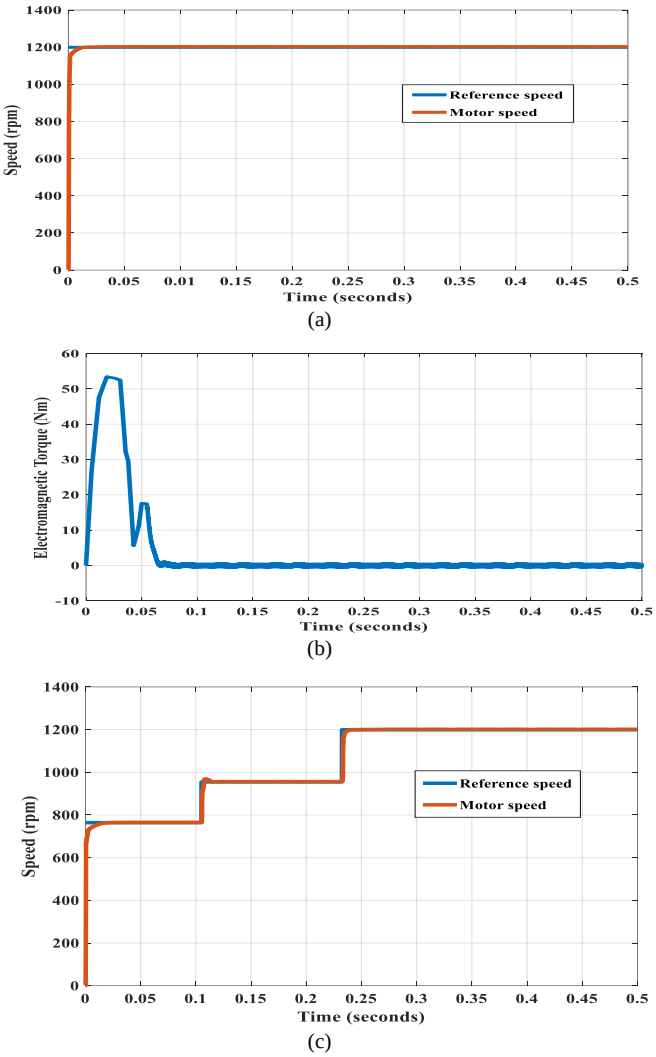


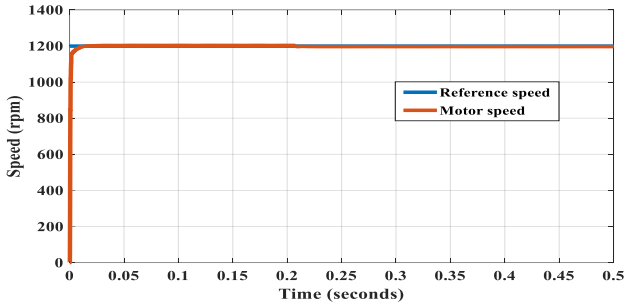
Fig.12. Response of the system with Fuzzy-PI controller without load

- With load

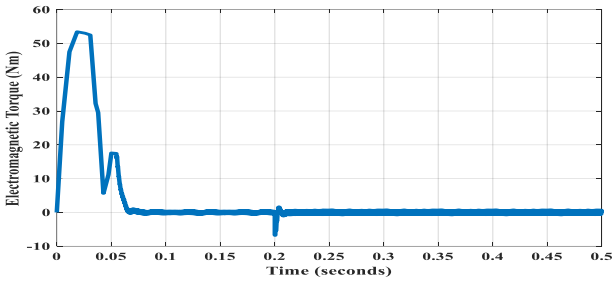
Upon the application of a 6.45 N.m load torque at 0.2 seconds, the system's speed-time response, as depicted in figure (13) (a), (b), and (c), demonstrates the robustness of the fuzzy-PI controller. The controller effectively restores the system to the desired trajectory, exhibiting a settling time of 0.0824 seconds, a rising time of 0.0540 seconds, and a steady-state error of 2.0186%. It is observed that an initial, instantaneous steady-state error arises at zero torque, impacting the system's performance. Furthermore, the fuzzy-PI controller demonstrates its capability to mitigate electromagnetic torque fluctuations and manage nonlinear behavior.

Specifically, the application of the 6.45 N.m load torque results in a reduction of the electromagnetic torque from 55N.m to zero N.m under no-load conditions and from 49N.m to zero N.m under loaded conditions, as illustrated in figures (12) and (13), respectively. This reduction highlights the controller's ability to effectively counteract the load disturbance and maintain system stability.

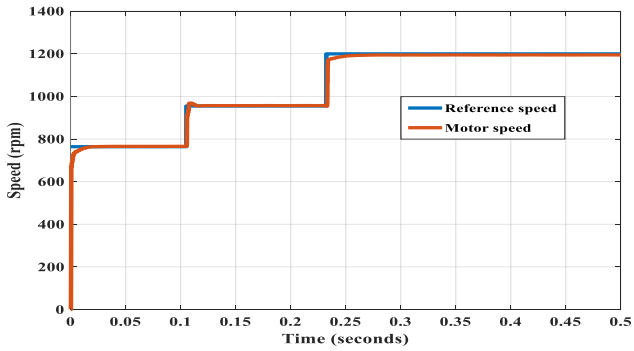
When the setpoint was varied by 764 rpm, 955 rpm, and 1200 rpm, the overshoot, rise time, settling time, and steady-state error were as shown in Table (7).



(a)



(b)



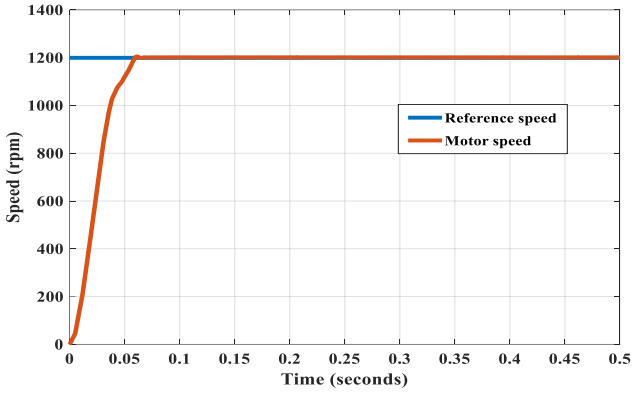
(c)

Fig.13. Response of the system with Fuzzy-PI controller with load

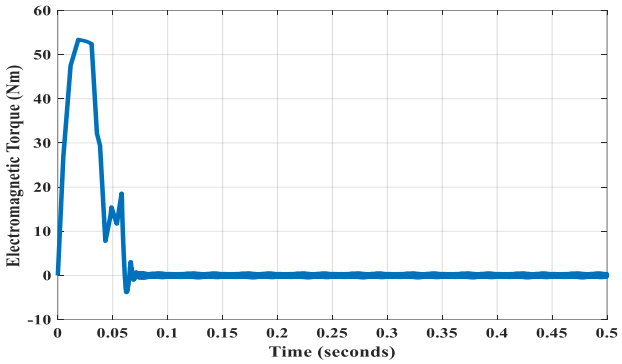
II. *The results that are obtained by applying the PI-controller*

- Nominal case

Figure (14) (a), and (b), illustrates the closed-loop system's response under no-load conditions, employing unity feedback. The output speed profile demonstrates a transient response characterized by a 0.3288 overshoot, a 0.0363-second rise time, and a 0.0567-second settling time. Furthermore, a 1.7525% steady-state error is observed. However, the response exhibits non-linear behavior, indicating limitations in the system's ability to fully compensate for inherent non-linearities, as further evidenced in figure (14).



(a)



(b)

Fig.14. Response of the system with PI-controller without load

- with load

Upon the application of a 6.45 N.m load, the system's speed-time response, as presented in figure (15) (a), and (b), reveals the limitations of the proportional-integral (PI) controller. While the controller demonstrates an attempt to restore the speed to the reference trajectory, its performance is characterized by significant deviations, indicating suboptimal regulation. Furthermore, the introduction of the load induces nonlinear behavior, which demonstrably affects the system's dynamic response, as illustrated in figure (15). Specifically, the observed settling and rising times are 0.0707 seconds and 0.0513 seconds, respectively, with a steady-state error of 0.5%. These performance metrics highlight the PI controller's inability to effectively mitigate the impact of the load and suppress the resulting nonlinearities.

The application of the 6.45 N.m torque also induces a substantial decrease in the electromagnetic torque. Under no-load conditions, the electromagnetic torque diminishes from 53 N.m to zero, while under load, it decreases from 49 N.m to zero, as evidenced in figures (14) and (15). This torque reduction underscores the controller's struggle to maintain stable operation in the presence of significant load disturbances and nonlinearities.

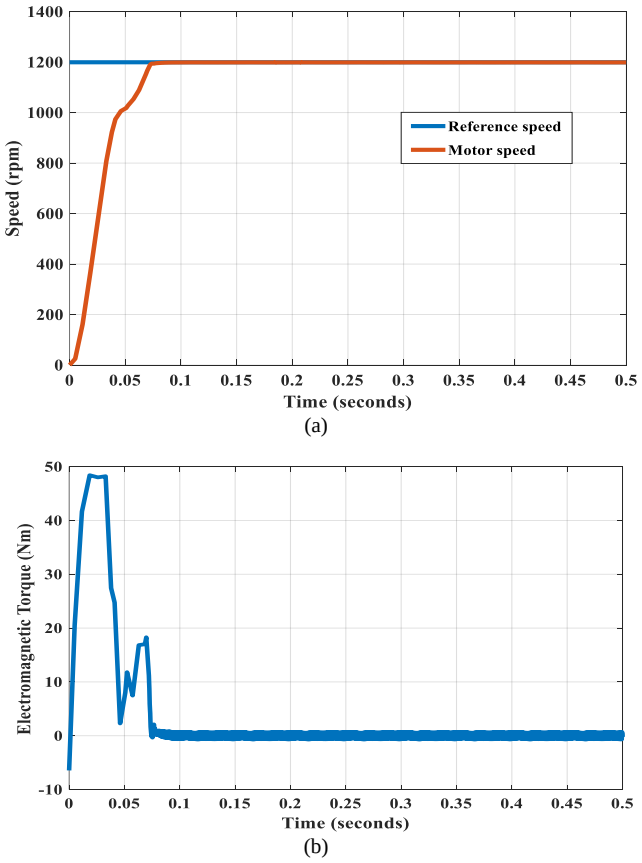


Fig.15. Response of the system with PI-controller with load

III. The results that are obtained by applying the suggested Neural Network (NN) controller

- Nominal case

Under no-load conditions, the closed-loop speed response with unity feedback, as depicted in figure (16) (a), (b), and (c), demonstrates a characteristic performance. The system achieves the reference speed with zero overshoot, a rise time of 0.0457 seconds, and a settling time of 0.0787 seconds. However, a steady-state error of 9.7098% is observed. In contrast, the application of a neural network controller effectively mitigates electromagnetic torque, reducing it from 53 N.m to zero without introducing non-linear behavior. This highlights the potential of neural network control in enhancing system performance and reducing steady-state errors, even in the absence of external loads.

When the setpoint was varied by 764 rpm, 955 rpm, and 1200 rpm, the overshoot, rise time, settling time, and steady-state error were as shown in Table (8).

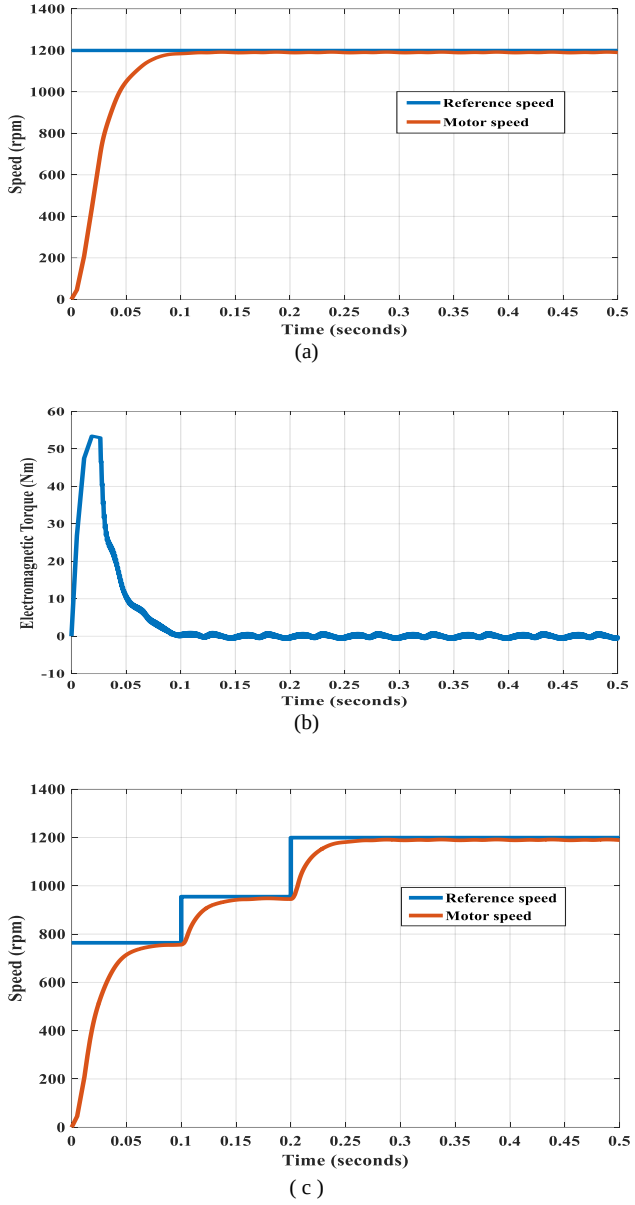
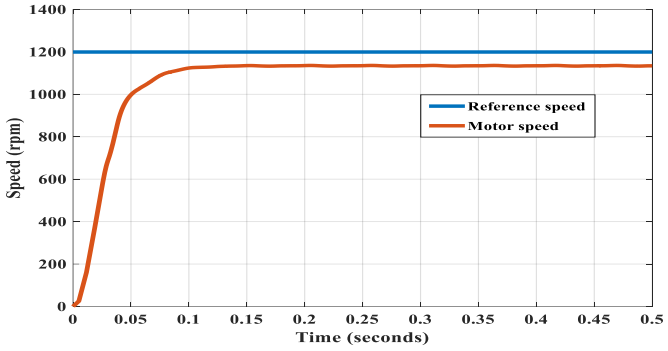


Fig.16. Response of the system with Neural Network controller without load

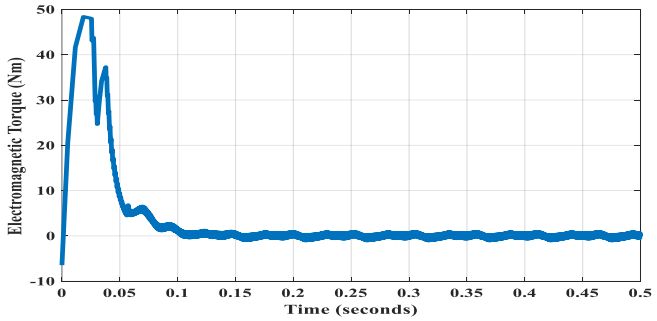
- With load case

When a 6.45 N.m load is applied, see figure (17) (a), (b), and (c), NN controller attempts to regulate the speed to the intended setpoint, it is clear that the controller performs poorly. Also, when the load is applied, the non-linear behavior affects the system behavior, as shown in figure (17). The settling and rising times are around (0.0894) and (0.0462) seconds respectively, whilst the steady-state error is (65.1070%). Furthermore, the NN controller couldn't reduce the electromagnetic torque and overcome nonlinear behavior. When the load is applied, it causes the electromagnetic torque to drop from 48 to Zero N.m with load, as illustrated in figure (16) and figure (17).

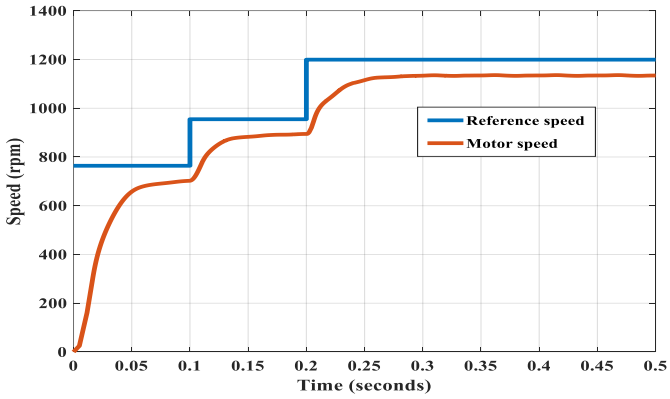
When the setpoint was varied by 764 rpm, 955 rpm, and 1200 rpm, the overshoot, rise time, settling time, and steady-state error were as shown in Table (9).



(a)



(b)



(c)

Fig.17. Response of the system with Neural Network controller with load

TABLE 4: The time domain specification of the system with load

Time domain specification	Controller With load		
	FLC	*NN*	*PI*
Rise Time	0.0540	0.0462	0.0513
Settling Time	0.0824	0.0894	0.0707
Overshoot	0.0156	0.1512	0.0107
Peak Time	0.4630	0.4696	0.4602
Error Steady-State (%)	2.0186	65.1070	0.5

TABLE 5: The time domain specification of the system without load

Time domain specification	Controller Without load		
	FLC	*NN*	*PI*
Rise Time	0.0392	0.0457	0.0363
Settling Time	0.0572	0.0787	0.0567
Overshoot	0.0243	0	0.3288
Peak Time	0.1268	0.4398	0.0611
Error Steady-State (%)	3.7987	9.7098	1.7525

TABLE 6: The time domain specification that are obtained by applying the proposed Fuzzy-PI controller without load

Time domain specification	Reference speed (rpm)		
	764	*955*	*1200*
Rise Time (sec)	0.078	0.1075	0.0392
Settling Time (sec)	0.082	0.151	0.0572
Overshoot (%)	0	1	0.0243
Error Steady-State (%)	1.9	1.8	3.7987

TABLE 7: The time domain specification that are obtained by applying the proposed Fuzzy-PI controller with load

Time domain specification	Reference speed (rpm)		
	764	*955*	*1200*
Rise Time (sec)	0.0308	0.0933	0.0540
Settling Time (sec)	0.0417	0.155	0.0824
Overshoot (%)	0	0	0.0156
Error Steady-State (%)	5.3	4.9	2.0186

TABLE 8: The time domain specification that are obtained by applying the suggested Neural Network (NN) controller without load

Time domain specification	Reference speed (rpm)		
	764	*955*	*1200*
Rise Time (sec)	0.092	0.045	0.0457
Settling Time (sec)	0.125	0.075	0.0787
Overshoot (%)	0	0	0
Error Steady-State (%)	4.5	8.7	9.7098

TABLE 9: The time domain specification that are obtained by applying the suggested Neural Network (NN) controller with load

Time domain specification	Reference speed (rpm)		
	764	*955*	*1200*
Rise Time (sec)	0.1032	0.0667	0.0462
Settling Time (sec)	0.1343	0.0978	0.0894
Overshoot (%)	0	0	0.1512
Error Steady-State (%)	61.2	63.9	65.1070

IV. **Conclusion**

A fuzzy-PI controller was designed for PMSMs drive and the obtained results compared with the traditional PI. The considered intelligent controller, which is the NN (Neural Network) controller, is studied and simulated in terms of

performance. In which MATLAB/SIMULINK environment is used for design, implementation, and testing. The speed and torque of the PMSMs drive were controlled satisfactorily. Simulation results have shown decent results for the two scenarios; with and without load. The fuzzy-PI controller demonstrated better performance in case of loading scenario than the PI and Neural Network controllers. This can be seen from the non-linear behavior of the other two (NN and PI) controllers; they were not able to overcome nonlinearity electromagnetic torque successfully.

REFERENCES

[1] Khanh, Pham Quoc, and Ho Pham Huy Anh. "Advanced PMSM speed control using fuzzy PI method for hybrid power control technique." *Ain Shams Engineering Journal* 14.12 (2023): 102222. DOI: https://doi.org/10.1016/j.asej.2023.102222.

[2] Zhang, Lei, Jing Bai, and Jing Wu. "Speed Sensor-Less Control System of Surface-Mounted Permanent Magnet Synchronous Motor Based on Adaptive Feedback Gain Supertwisting Sliding Mode Observer." *Journal of Sensors* 2021.1 (2021): 8301359. DOI: https://doi.org/10.1155/2021/8301359.

[3] Vu Quynh, Nguyen. "The fuzzy PI controller for PMSM's speed to track the standard model." *Mathematical Problems in Engineering* 2020.1 (2020): 1698213. DOI: https://doi.org/10.1155/2020/1698213.

[4] Liu, Xin, et al. "Active Disturbance Rejection Sensorless Control of Permanent Magnet Synchronous Motor Based on the Fuzzy Neural Network Left Inverse System." *Progress in Electromagnetics Research C* 130 (2023).

[5] Zeng, Xiaoli, Weiqing Wang, and Haiyun Wang. "Adaptive PI and RBFNN PID current decoupling controller for permanent magnet synchronous motor drives: Hardware-validated results." *Energies*15.17 (2022): 6353. DOI: https://doi.org/10.3390/ en15176353.

[6] Zhang, Qi, and Caiyue Zhang. "Speed Control of PMSM Based on Fuzzy Active Disturbance Rejection Control under Small Disturbances." *Applied Sciences* 13.19 (2023): 10775. DOI: https://doi.org/ 10.3390/app131910775.

[7] Hamdi Pacha, Fatima Zohra, et al. "Intelligent monitoring of PMSM based on adaptive fuzzy logic for diagnosis." *Przeglad Elektrotechniczny* 99.4 (2023). DOI:10.15199/48.2023.04.42.

[8] Sahridayan, Parvathy Thampi Mooloor, and Raghavendra Gopal. "Modeling and analysis of field-oriented control based permanent magnet synchronous motor drive system using fuzzy logic controller with speed response improvement." *International Journal of Electrical and Computer Engineering (IJECE)* 12.6 (2022): 6010-6021. DOI: 10.11591/ijece. v12i6.pp6010-6021.

[9] VO, Hau Huu, and Pavel BRANDSTETTER. "Modified Fuzzy Logic PI Speed Controller with Scheduling Boundaries of Integral Time Constant for PMSM Drive." *Przeglad Elektrotechniczny* 2023.11 (2023). DOI:10.15199/48.2023.11.30.

[10] Fayek, Haytham M., and I. Elamvazuthi. "Type-2 fuzzy logic PI (T2FLPI) based dc servomotor control." *Journal of Applied Sciences Research* 8.5 (2012): 2564-2574.

[11] Tummala, Suresh Kumar, and G. Dhasharatha. "Artificial neural networks based SPWM technique for speed control of permanent magnet synchronous motor." *E3S web of conferences*. Vol. 87. EDP Sciences, 2019. DOI: https://doi.org/10.1051/e3sconf/2019870103.

[12] Jantzen, Jan. "Tutorial on fuzzy logic." *Technical University of Denmark, Dept. of Automation, Technical Report* (1998).

[13] Kart, Diagram Source Brain. "Pulse width modulation." (2001).

[14] عبدالكريم حدود, et al. "The Application of An Intelligent Adaptive Controller for Permanent Magnet Synchronous Motor Drive Using Neural Network."586-574 :(2021) 6.5 مجلة الجامعة الأسمرية.