An International Journal of Optimization and Control: Theories & Applications
ISSN: 2146-0957 eISSN: 2146-5703
Vol.15, No.3, pp.435-448 (2025)
https://doi.org/10.36922/IJOCTA025070023

RESEARCH ARTICLE

FPGA design and implementation of fuzzy learning control: Application on DC motor position control

Mohand Achour Touat[1], Hocine Khati[1], Arezki Fekik[2,3], Ahmad Taher Azar[4,5], Hand Talem[1], Rabah Mellah[1] and Saim Ahmed[4,5*]

[1]Design and Drive of Production Systems Laboratory, Faculty of Electrical and Computing Engineering, University Mouloud Mammeri of Tizi-ouzou, Tizi-ouzou, Algeria
[2]Nantes University, Centrale Nantes, CNRS, LS2N, UMR 6004, Nantes, France
[3]Department of Electrical Engineering Faculty of Applied Sciences, University of Bouira, Bouira, Algeria
[4]College of Computer and Information Sciences, Prince Sultan University, Riyadh, Saudi Arabia
[5]Automated Systems and Computing Lab (ASCL), Prince Sultan University, Riyadh, Saudi Arabia
Mohand-achour.touat@ummto.dz, hocine.khati@ummto.dz, Arezki.Fekik@univ-nantes.fr, aazar@psu.edu.sa, talemhand2015@gmail.com, Rabah.mellah@ummto.dz, sahmed@psu.edu.sa

ARTICLE INFO

Article History:
Received: February 10, 2025
Revised: March 12, 2025
Accepted: March 27, 2025
Published Online: May 8, 2025

Keywords:
Learning fuzzy control
FPGA
HDL Coder
Adaptive control
FIL
Optimization

AMS Classification 2010:
26A33; 34A08; 35H15; 34K50 47H10; 60H10

ABSTRACT

This paper investigates the implementation of a Fuzzy Model Reference Learning Control (FMRLC) on a Zedboard Zynq-7000 FPGA. The proposed adaptive controller dynamically adjusts its knowledge base and incorporates a memory-based control mechanism to retain and utilize past results in recurring situations. The design and deployment of the controller were carried out using the MATLAB/Simulink environment and applied to the angular position control of a DC motor. Initially, the controller was tested using the FPGA-In-the-Loop (FIL) approach to assess its robustness against disturbances in simulation. Subsequently, it was experimentally validated for real-time motor position control. The results obtained in FIL simulations and experimental tests demonstrate high tracking accuracy and strong disturbance rejection. These findings underscore both the superiority of the proposed controller over the conventional PID controller and the effectiveness of the adopted design methodology.

1. Introduction

In recent years, fuzzy logic applications have experienced significant growth across various domains, ranging from industrial applications such as medical instruments and robotics to consumer products like washing machines, cameras, and automobiles. Fuzzy logic, based on fuzzy set theory, offers a set of if-then rules, eliminating the need for complex differential equations. This approach provides a structured methodology for modeling, manipulating, and implementing human heuristic knowledge in control systems. Widely adopted in automatic control, fuzzy logic avoids intricate nonlinear equations while leveraging expert knowledge. However, experts cannot always anticipate uncertainties arising during system operation. To ensure safe system functionality and accurate trajectory tracking despite disturbances, it is crucial to automatically update human knowledge without operator intervention.

To address these challenges, various adaptive control algorithms have been introduced in the literature, including fuzzy adaptive control.[1–4]

*Corresponding Author

While adaptive control can update the knowledge base, it may yield inconsistent results for identical uncertainties or disturbances.[5–8] To overcome this limitation, learning adaptive control strategies have been proposed, equipping control mechanisms with memory. These methods ensure consistent results for recurrent situations, thereby reducing computation time.[9–12]

Although fuzzy controllers demonstrate excellent performance in simulation, their practical implementation demands high-performance control boards due to significant computational requirements. Field Programmable Gate Arrays (FPGAs) have emerged as a powerful solution in control applications,[13] enabling high sampling frequencies, parallel data processing, and low power consumption.[14] Their integration accelerates control algorithms, ensuring rapid convergence toward desired performance.[15] Several studies have explored FPGA-based control implementations. For instance, Huerta et al.[16] examined FPGA implementation of PID and sliding mode controllers for DC-DC buck converters. Wang et al.[17] developed a neural network-based PID controller for motion control systems. Similarly, Li et al.[18] investigated FPGA-accelerated predictive control for autonomous driving. Meghwal et al. [19] presented a robust fuzzy controller implemented on FPGA for matrix converter-based induction motor control, while Attafi et al.[20] focused on real-time FPGA implementation of robust control for wind energy conversion systems.

In this paper, a learning adaptive control strategy, namely the Fuzzy Model Reference Learning Control (FMRLC), is implemented on a Zedboard Zynq-7000 FPGA for DC motor position control. The proposed controller is first validated in a real-time simulation environment using the FPGA-in-the-Loop (FIL) approach, a technique widely employed in laboratories to assess prototype controllers under real operating conditions. Compared to conventional numerical simulations, FIL provides a more realistic validation by incorporating real-world factors such as noise, disturbances, and implementation challenges, which may not be accounted for in idealized simulations.

The proposed methodology leverages MATLAB/Simulink for controller design and FPGA implementation. Simulink, in conjunction with HDL Coder, enables direct FPGA deployment without requiring VHDL programming.[21] However, since floating-point data types in Simulink are not supported by all FPGA platforms, the Fixed-Point Tool is utilized to convert floating-point designs into fixed-point representations. This conversion optimizes FPGA hardware resource utilization and minimizes power consumption.

Several implementation works on FPGA based on fuzzy logic have been carried out. [22–31] In,[22,23] the implementation of a fuzzy controller on FPGA using the FIL technique on the MATLAB-Simulink environment is presented. The design, simulation, and FPGA implementation of a fuzzy controller to regulate the speed of a DC motor in real time is presented in.[27] In,[28] genetic and pipeline algorithms are implemented in FPGA to optimize fuzzy controller parameters. In ref., [29] an FPGA design approach with partial reconfiguration (PR) using a fuzzy controller is used to control an electromechanical system.

This study presents the implementation of an FMRLC-based position control strategy for a DC motor, incorporating a memory-based control mechanism to ensure consistent responses to recurring conditions, thereby reducing computational overhead. In the first phase, the fuzzy controller is designed in MATLAB/Simulink and implemented on a Zedboard Zynq-7000 FPGA using the FIL technique. In this setup, the FMRLC controller runs on the FPGA, while the remaining control system components operate within Simulink to assess performance and robustness under various disturbances. The second phase involves deploying the algorithm onto the FPGA board using HDL Coder's "IP Core Generation" technique for real-time implementation. This method allows direct integration of the control strategy onto the Zedboard by generating an IP block via Vivado software, eliminating the need for manual VHDL coding and significantly reducing development time. "IP Core Generation" automatically generates the VHDL code, compiles the bitstream programming file, and transfers it to the Zedboard.

The structure of this paper is as follows: Section 2 provides an overview of the FMRLC control strategy. Section 3 details the mathematical modeling of the DC motor in both theoretical and experimental contexts. Section 4 describes the design and FPGA implementation of the FMRLC controller. Section 5 presents simulation results in FIL mode along with experimental validation. Finally, the conclusions are summarized in Section 6.

2. Architecture of the control strategy

The Fuzzy Model Reference Learning Controller (FMRLC) is shown in Figure 1. As illustrated , it has three main parts as defined by[32]: Fuzzy Direct Controller (FC) synthesized to control the DC

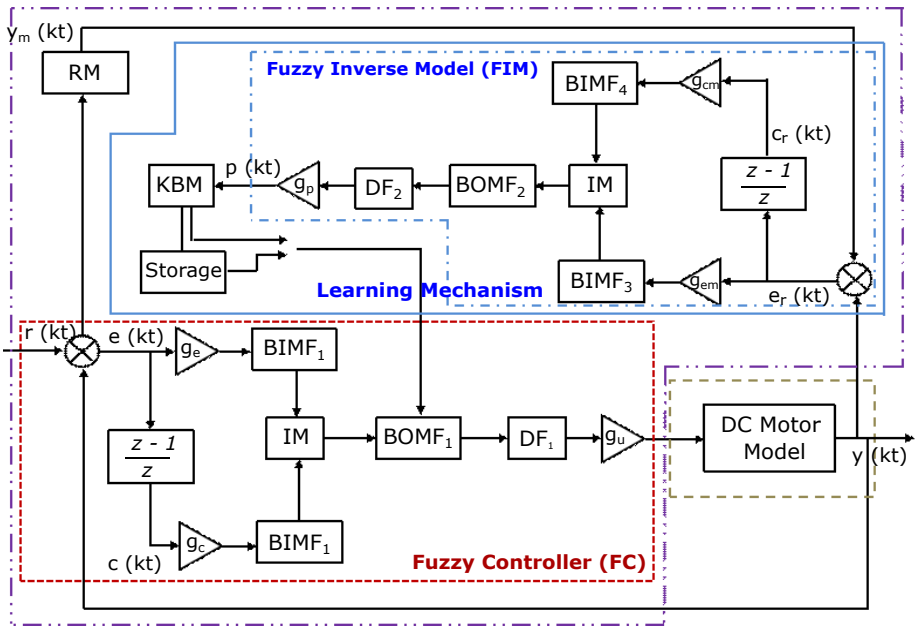

Figure 1. Block diagram of FMRLC

motor, the reference model (RM), and the learning mechanism. This last part is divided into two subparts: Knowledge Base Modifier (KBM) and Fuzzy Inverse Model (FIM). This control strategy is based on the Model Reference Adaptive Direct Control (MRAC). This method aims to synthesize a fuzzy controller, by adjusting its membership functions in order to reject different disturbances of the controlled system. The main parts of the FMRLC control scheme are detailed in Figure 1 and are described as follows:

2.1. Direct fuzzy logic controller

The Fuzzy Controller (FC) designed in this paper is to generate a control signal in order to force the output of the system to follow the assigned DC motor position reference signal; it has two inputs:

$$e\,(kt) = r\,(kt) - y\,(kt) \tag{1}$$

$$c\,(kt) = \frac{z-1}{z} e\,(kt) \tag{2}$$

$e\,(kt)$ defines the error between the position reference signal and the position output signal of the DC motor, $c\,(kt)$ constitutes the error change. These inputs are normalized by the mean of the scaling factors g_e and g_c that belong to the interval [-1 1]. The input membership functions of the error and its derivatives are introduced, respectively, in BIMF1 and BIMF2 with the number 5, while the output ones are given in the BOMF1 with the number 5. The inference mechanism (IM) used in this study is of Mamdani type based on the min-max inference method, and is expressed in the form of IF-THEN rules. Its rule base is summarized in Table 1. The membership functions are shown in Figure 2, with five fuzzy sets on the universe of discourse, with linguistic values, which are the following: Negative Big (NB), Negative (N), Zero (ZE), Positive (P), and Positive Big (PB).

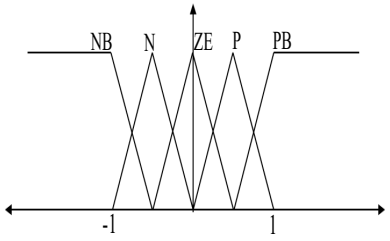

Figure 2. Membership functions

Table 1. Initial rule base of the fuzzy controller

		$c\,(kt)$	$c\,(kt)$	$c\,(kt)$	$c\,(kt)$	$c\,(kt)$
		-1	-0.5	0	0.5	1
$e\,(kt)$	-1	-1	-1	-1	-0.5	0
$e\,(kt)$	-0.5	-1	-1	-0.5	0	0.5
$e\,(kt)$	0	-1	-0.5	0	0.5	1
$e\,(kt)$	0.5	-0.5	0	0.5	1	1
$e\,(kt)$	1	0	0.5	1	1	1

The defuzzification part is introduced in DF1. In this case, the Center Of Gravity (COG) method is used. The output of FC constitutes the control signal $u\,(kt)$ and normalized to the same interval as the input with the scaling factor g_u. The dynamic model of the DC motor is approximated with the first-order model and represented in the Reference Model (RM).

The last part of the block diagram shown is the learning mechanism; this part modifies the knowledge base with respect to plant parametric disturbances. The latter modifications are made on the output membership functions in order to force the closed-loop system to behave like the reference model. According to Figure 1, the learning mechanism consists of following two parts: a fuzzy inverse model (FIM) and a knowledge base modifier (KBM). The fuzzy inverse model is based on the same inference mechanism as the fuzzy controller (FC). The inputs of the FIM are as follows:

$$e_r(kt) = y_m(kt) - y(kt) \tag{3}$$

$$c_r(kt) = \frac{z-1}{z} e_r(kt) \tag{4}$$

$e_r(kt)$ is the error between the output of the reference model and the plant output; $c_r(kt)$ is the error change. The output of this controller is the necessary changes in the plant inputs. Likewise to the fuzzy controller, the FIM shown in Figure 1 contains normalizing scaling factors, namely g_{em}, g_{cm} and g_p for each universe of discourse. It is important to notice that the selection of the normalizing gains can impact the overall performance of the system. For this reason, it is essential to use an optimization algorithm to compute these parameters. This is done by using the MATLAB/Simulink response optimization tool. The output of the FIM represents the variation of the FC's output membership function center. In order to get a better performance, modification of the FC rule base is performed by the knowledge base modifier (KBM).

2.2. Reference model

The reference model is chosen so that the system will have a rapid response. In this case, the choice is made for a model with the best dynamics ensured by conventional control under perfect conditions, i.e., no disturbances or parametric variations. This dynamic is approximated by the following first-order model:

$$R(s) = \frac{1}{1+\tau_m s} \tag{5}$$

Where τ_m is the time constant to be determined.

2.3. Learning mechanism

The learning mechanism adjusts the ground rules of the direct FC by exploiting information such as the output of the controlled system $y(kt)$ and that of the reference model and the already existing basis of the direct FC.

The output $p(kt)$at instant $(kt-t)$ represents the correction to be made to the command at instant in order to reduce the following error $e_r(kt)$. In other words, we force the FC to produce the desired command: $u(kt-t)+p(kt)$ necessary to cancel the tracking error $e_r(kt)$. So, the next time we encounter the same circumstances, the command will be the one that will reduce the tracking error.

3. Mathematical model of the DC motor

In order to be able to simulate and implement the closed-loop systems on the FPGA board, the mathematical model of the DC motor is required. The equivalent electric circuit of the DC motor armature and the diagram of the rotor are shown in Figure 3.

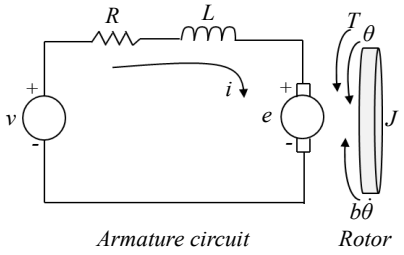

Figure 3. Electrical circuit of a DC motor

Let us take the input of the system the voltage source v applied to the motor armature, while the output is the position of the shaft θ.

Generally, the torque generated by the DC motor is proportional to the armature current and the magnetic fields. Suppose that the latter is constant; therefore, the torque T is proportional to the current of the armature i by a constant factor K_1:

$$T = K_1 i \tag{6}$$

The electromotive force E is proportional to the velocity of the shaft $\dot{\theta}$ by a constant factor K_2:

$$E = K_2 \dot{\theta} \tag{7}$$

The constants of the torque T and the electromotive force E are equal, i.e., $K_1 = K_2$. Therefore, we take $K_1 = K_2 = K$.

From the Figure 3, we have the following equations:

$$J\ddot{\theta} + b\dot{\theta} = Ki \tag{8}$$

$$L\frac{di}{dt} + Ri = v - K\dot{\theta} \tag{9}$$

By applying the Laplace transform to the above equations, we obtain :

$$(Js + b)\, s\theta(s) = KI(s) \tag{10}$$

$$(Js + b)\, I(s) = V(s) - Ks\theta(s) \tag{11}$$

From Equation (10) , the current formula is given by:

$$I(s) = \frac{(Js + b)\, s\theta(s)}{K} \tag{12}$$

From Equations (11) and (12), and considering that the output of the system is the angular position of the motor shaft $\theta(s)$, and the input of the system is the voltage of the armature $V(s)$, the transfer function of the DC motor is :

$$\frac{\theta(s)}{V(s)} = \frac{K}{s\left[(Ls + R)(Js + b) + K^2\right]} \tag{13}$$

Where :
J : Moment of inertia of the rotor
R : Electric resistance
L : Electric inductance
b : Motor viscous friction constant
However, in our case, the numerical values of the above constants are not available, so the calculation of the model of the DC motor must be done experimentally using the Zedboard FPGA.

The DC motor is considered as a black box. Its identification consists of applying a voltage V and obtaining the angular position θ of the motor. Based on these measurements, the MATLAB system identification toolbox[33] is used to calculate the parameters of the motor model.

During the identification process, it is discovered that the DC motor does not start within a range of voltage values. Knowing that the motor supply voltage is 12V, the dead zone of this motor is [-2.4V 2.4V]. The model obtained by the System Identification Toolbox is represented by the following discrete transfer function, with a sampling time of T_e=0.001 s.

$$\begin{aligned} G(z) = \frac{\theta(z)}{V(z)} &= \frac{b_0}{a_2 z^2 + a_1 z + a_0} \\ &= \frac{16.4}{z^2 + 0.6615z + 0.00041255} \end{aligned} \tag{14}$$

4. FPGA implementation

Before applying the controller to the experimental device, we must first test its behavior on the Zedboard Zynq-7000 FPGA using the FPGA-in-the-loop (FIL) simulation technique in the Simulink environment of MATLAB.

4.1. Fixed point data type

FPGA circuits generally support the following two types of data: fixed-point data and floating-point data. However, designs based on the fixed-point data type consume fewer hardware resources and less power, with shorter computation times.[34]

However, most complex algorithms have nonlinear functions that are not supported by the fixed-point data type. These functions must, therefore, be approximated by other linear functions, which negatively affects the accuracy of the computations. In this work, the design methodology adopted is based on MATLAB-Simulink software.

The fixed-point representation on MATLAB-Simulink used in this work is as follows: $fixdt(s, l, d)$. $fixdt$ creates a numerical type on Simulink describing a fixed point. s represents the sign bit, l the length of the binary word and d the length of its fractional part.

For example, $fixdt(1, 16, 8)$: Here, 16 represents the length of the signal in bits, 1 is the sign bit (signed in this case), and 8 is the length of the fractional part in bits. Details of the fixed-point representation in MATLAB-Simulink are given in Table 2.

Table 2. Fixed-point représentation in Simulink

$Datatypemode$	$Fixed-point:$ $binarypointscaling$
$Signedness$	$Signed$
$Wordlength$	16
$Fractionlength$	8

The control algorithm is developed in the Simulink environment as a double-precision floating-point model using blocks supported by the HDL Coder and then converted to a fixed-point model using the Fixed-Point Tool. This tool calculates the correct integer and fractional parts of each input/output block from the developed floating-point model, respecting the rules of fixed-point arithmetic and optimizing binary word lengths.

4.2. FPGA in the loop simulation

Before applying it to the motor control, the proposed algorithm must be simulated on the MATLAB-Simulink environment with the system in order to test its behavior during various robustness tests. The proposed controller is developed on Simulink with the blocks of HDL Coder and converted into a model with a fixed-point data type, then its VHDL code is generated, which is

finally implemented on the hardware (FPGA) and then tested on the Simulink environment through the FIL functionality.[21] In this study, the Xilinx Zedboard is used, which has a Zynq-7000 AP SoC XC7Z020-CLG484-1 FPGA.[35]

FIL mode simulation is performed for real-time DC motor position control. The generated VHDL code is executed in the target hardware to control the DC motor model synthesized in Simulink. The Zedbaord exchanges the send data of position and signal control in each sampling period with the Simulink environment via the JTAG communication protocol. The concept of FIL simulation is described in Figure 4.

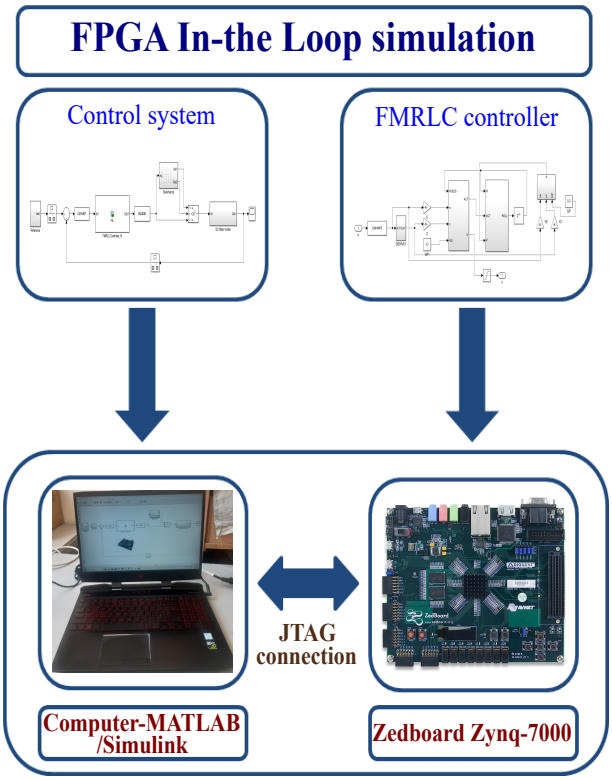

Figure 4. FPGA in-the loop diagram

4.3. Experimental implementation

4.3.1. Experimental setup

The control system consists of a 12 V DC motor (model DFRobot 28PA51G) equipped with a Hall Effect encoder and an adapter board for the encoder (FIT0324) Figure 5, which allows to recover the motor shaft position signal. The encoder can provide 675 pulses per revolution, with a maximum speed of 143 rpm. This motor is connected to an L298N (H-bridge) module, which will allow to change the speed and direction of the motor rotation.

The reference signal was generated by another encoder similar to the motor's. The controller, the two encoders, and the PWMs were programmed using Simulink blocks supported by HDL Coder. The encoder output was programmed as 16 bits, while the PWM input was programmed as 8 bits. The controller output was calculated using the Fixed-Point Tool by combining 16-bit integers and fractions to ensure accuracy. Figure 6 shows the control diagram developed in the Simulink environment.

This model was implemented via HDL Coder from Simulink, by "HDL Workflow Advisor" through the "IP Core Generation" functionality. This technique makes it possible to generate an IP block (Figure 7), whose target hardware is the zynq-7000 FPGA. Next, the VHDL code is generated from the IP block, which is then implemented on the Zedbaord as a bitstream programming file.

The implementation blocks obtained using the Vivado software are as follows:

The encoder block (Figure 8) has two logic inputs, A and B, representing the square-wave signals provided by the sensor, and a 16-bit integer output (int16) representing the position of the motor shaft (in pulses).

Figure 9 shows the PWM block, which includes an input u (the control signal) coded on 8 bits (unit8) and an output that is the DC motor supply voltage.

The FMRLC controller block (Figure 10) has an input e, which is the error between the desired position and the measured position, coded in fixed point on 16 bits ($fixdt(1,16,8)$) and an output u, which is the control signal, also coded in fixed point on 16 bits ($fixdt(1,16,6)$).

The PID controller block is shown in Figure 11, it has the position error e as an input and the control signal u as an output, both coded in fixed point to 16 bits (($fixdt(1,16,8)$) for e and ($fixdt(1,16,4)$) for u).

Details of the implementation of the PID and FMRLC controllers on the Zedboard FPGA are given in Table 3. These results are obtained in both cases using the 16-bit fixed-point data type.

The controller execution time depends on the controller complexity and the number of bits used for the signals. As can be seen, the PID has a lower propagation time than the FMRLC due to the large number of blocks it contains.

Table 3. Features of PID and FMRLC controller implementation on the Zedboard FPGA

Controller	Bits	Propagation	Maximum frequency
PID	16	14.14 (ns)	70.85 (MHz)
FMRLC	16	98.115 (ns)	10.19 (MHz)

4.3.2. FPGA resources

Despite the complexity of the algorithm, the use of the fixed-point tool allows to obtain an optimal

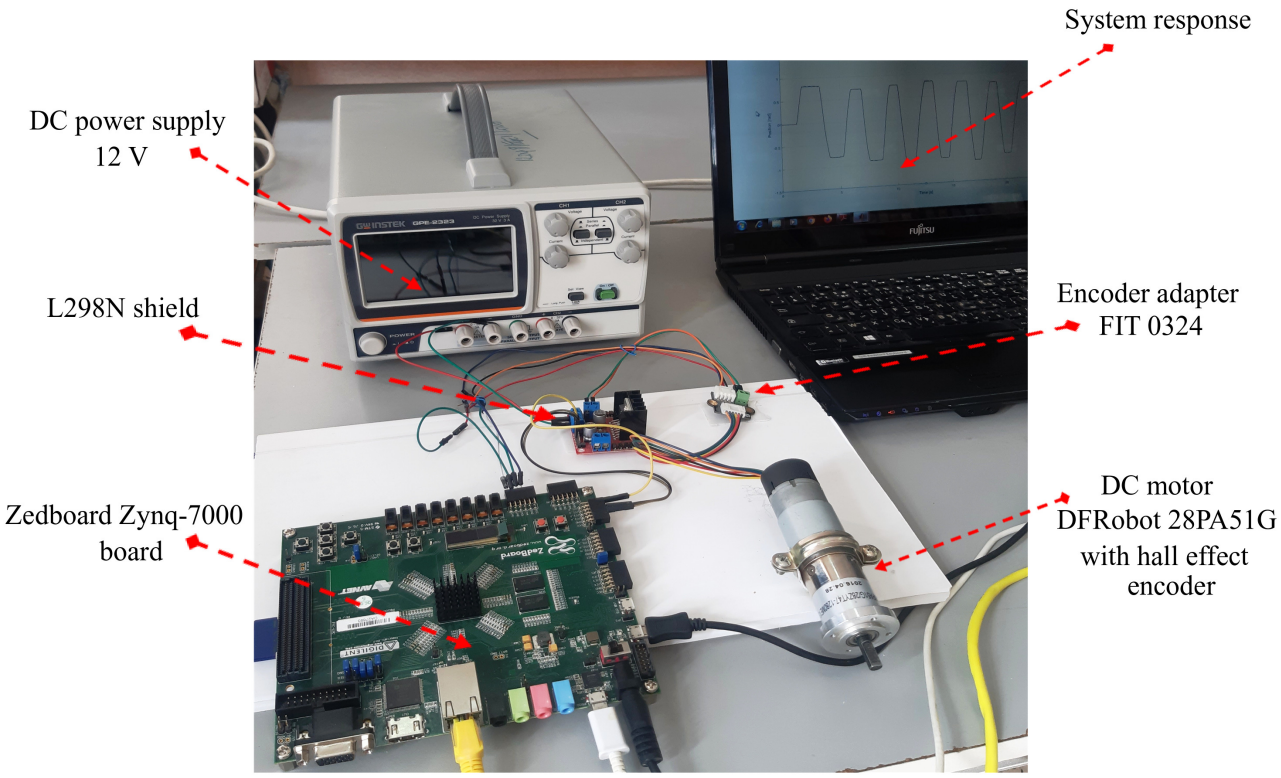

Figure 5. Experimental setup

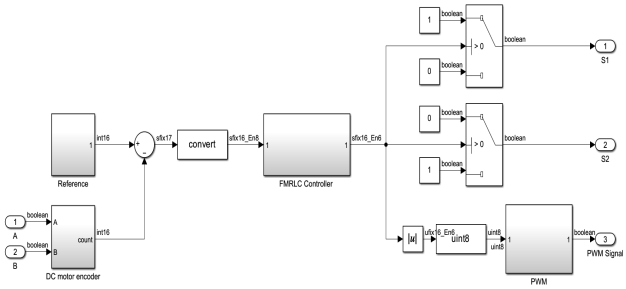

Figure 6. Control diagram in Simulink environment

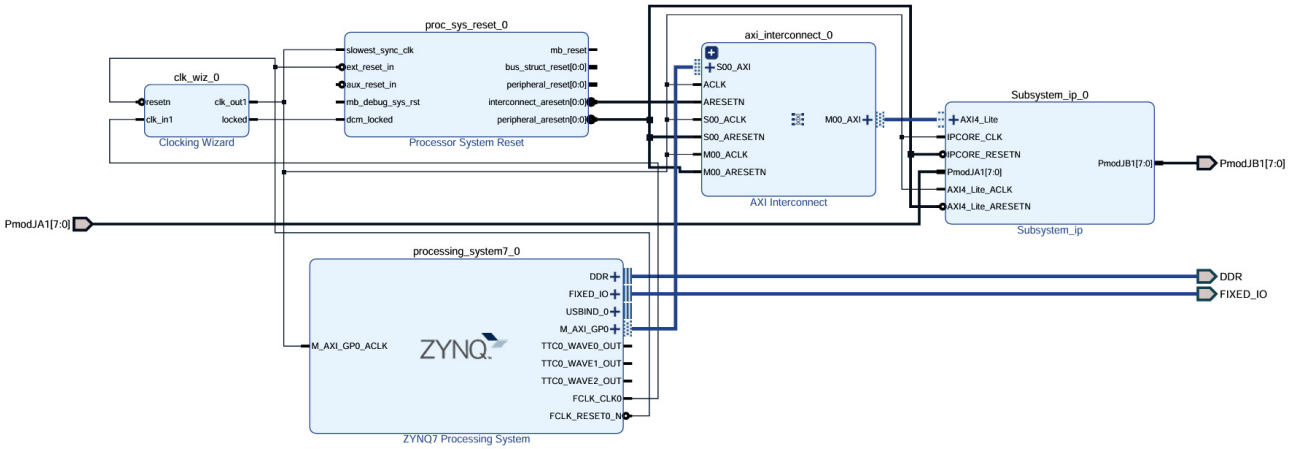

Figure 7. IP block of the realized design

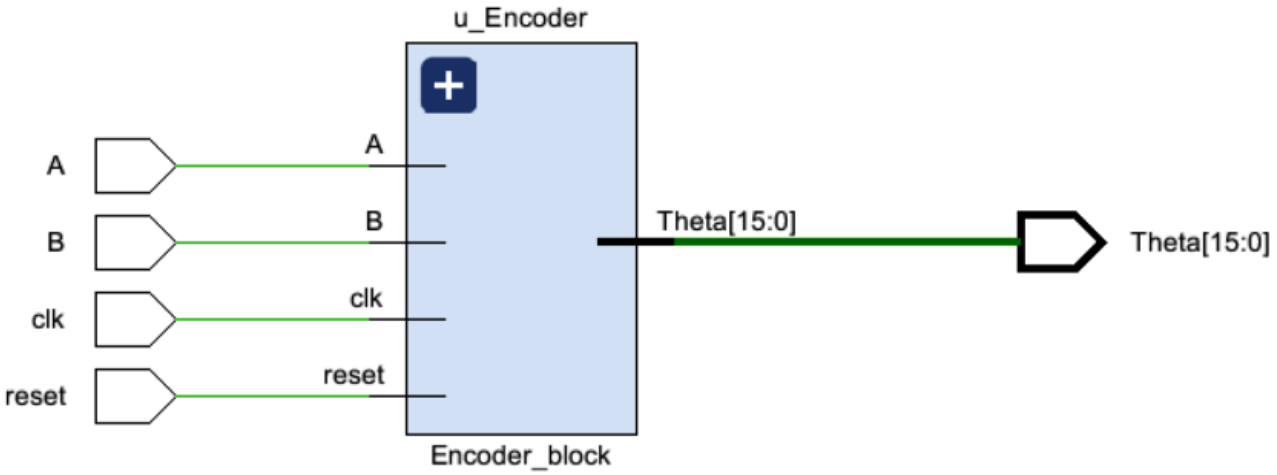

Figure 8. Encoder implementation block on FPGA

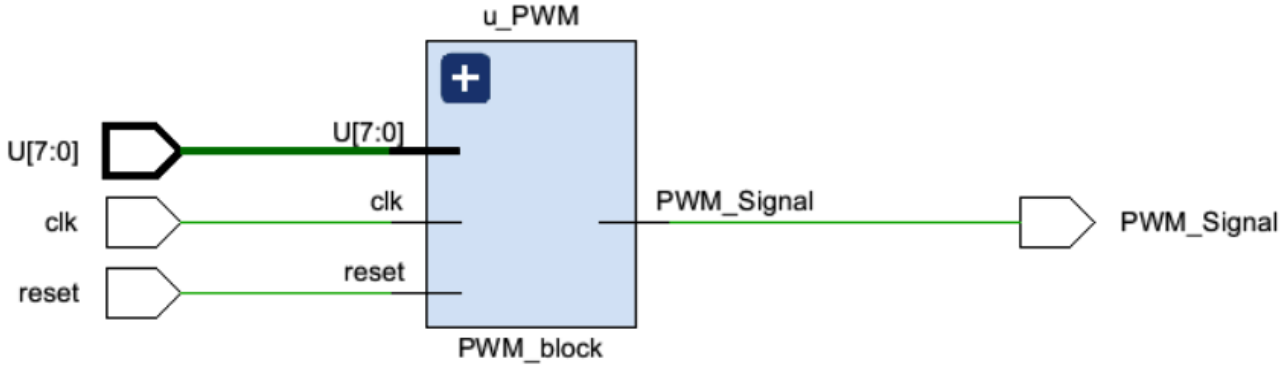

Figure 9. PWM implementation block on FPGA

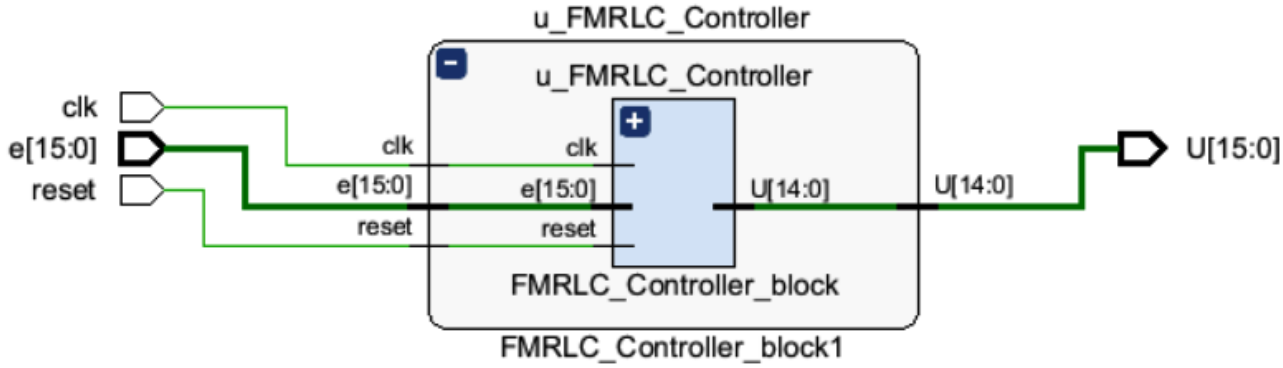

Figure 10. FMRLC implementation block on FPGA

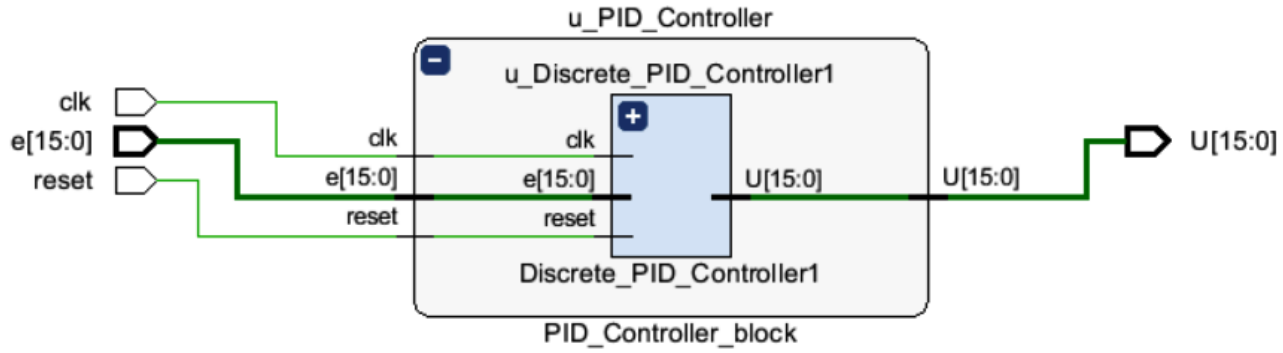

Figure 11. PID implementation block on FPGA

VHDL code in terms of computation time and resources consumed by the FPGA circuit.[36,37] Table 4 illustrates the hardware resources consumed when implementing the PID and FMRLC controllers on the Zedboard, using a sample rate of 1 MHz. As we can see, the FMRLC control consumed 12.68% of LUTs (Look Up Table), 0.47% of LUTRAMs (Memory LUT), 1.62% of FFs (Flip-Flop), 38.18% of DSPs (Digital Signal Processor), 6.00% of IOs (Input/Output), 6.38% of BUFGs (Global Buffer) and 25.00% of MMCMs (Mixed-Mode Clock Manager).

Table 4. FPGA resources consumption

Resource	Available	Utilization (PID)	Utilization (FMRLC)
LUT	53200	651(1.22%)	6801(12.68%)
LUTRAM	17400	39(0.22%)	81(0.47%)
FF	106400	664(0.62%)	1725(1.62%)
DSP	220	6(2.73%)	84(38.18%)
IO	200	12(6.00%)	12(6.00%)
BUFG	32	3(9.38%)	3(9.38%)
MMCM	4	1(25.00%)	1(25.00%)

5. Results

5.1. FIL simulation results

In this section, the results obtained by implementing the proposed algorithm with FPGA in the loop are presented in order to see the behavior of the proposed controller in case tracking with a sinusoidal reference. In this execution mode, the FMRLC controller is implemented and executed on the Zedboard with a sampling frequency of 1 kHz, while the rest of the control loop (DC motor model and reference) is executed in the Simulink environment (Figure 12).

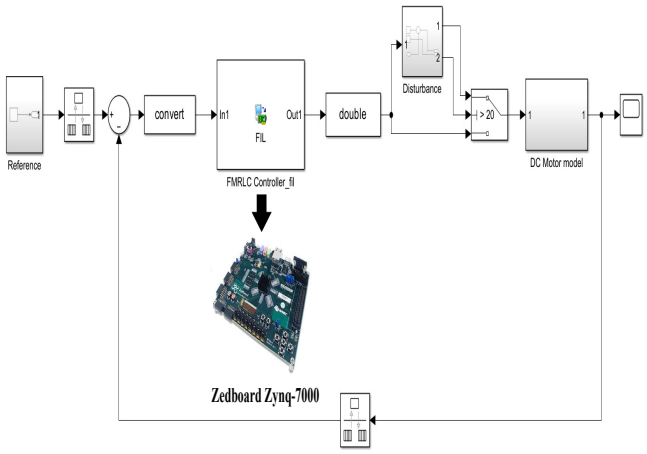

Figure 12. FMRLC controller implementation with FPGA In-the Loop

5.1.1. FIL simulation with parametric disturbances

In order to prove the efficiency and robustness of the proposed algorithm against parameter uncertainties, these parameters are varied in real-time simulation (see Table 5):

Table 5. DC motor model parameter variations

Time	t=8s	t=15s	t=23s	t=32s
Parameter	b_0	a_1	a_0	Dead zone
Value	10	2	2	[-3.1V 3.3V]

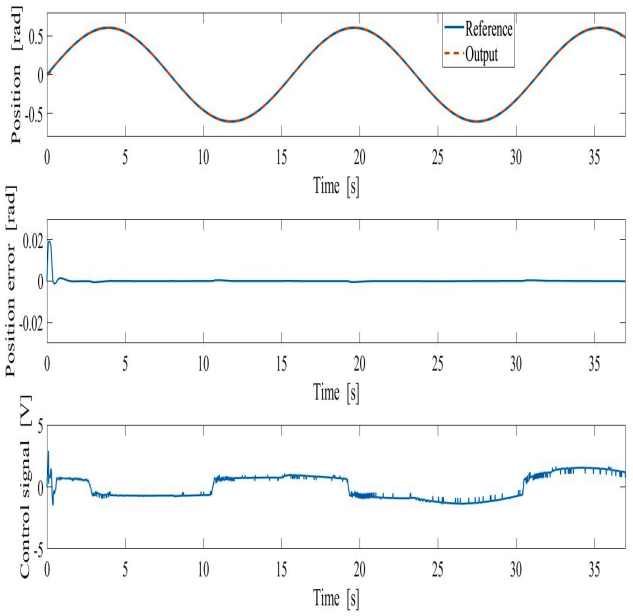

Figure 13. Simulation results of position tracking with parametric disturbances

The results obtained in Figure 13 show a good tracking of the reference changes with an error close to zero, despite the parametric variations made on the parameters of the motor model at times t = 8 s, t = 15 s, and t = 23 s, as well as the variation of the dead zone at time t = 32 s. The proposed controller is able to adapt to the dynamic variations of the system by modifying the parameters of the membership functions in a short time, thanks to the controller structure equipped with a learning mechanism with memory and to the high computational power of the FPGA.

5.1.2. FIL simulation with disturbances on the control

In this case, the robustness test is performed by injecting a disturbance at time t = 20s on the control signal provided by the FMRLC controller. The control signal provided by the Zedboard is 3.3 V, and the perturbation formula is (expressed in volts):

$$D(t) = 0.52sin(t) \tag{15}$$

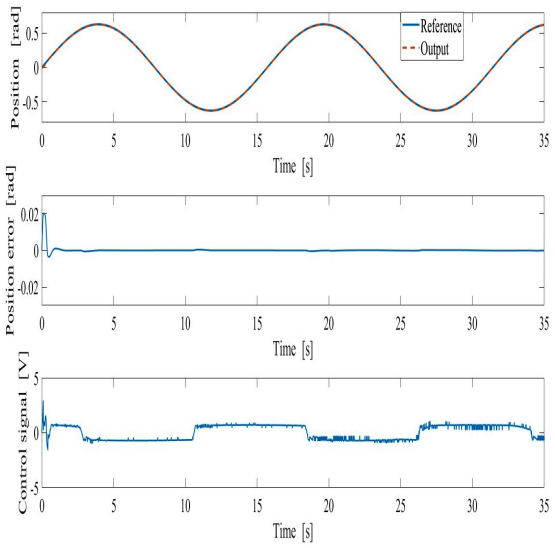

Figure 14. Simulation results of position tracking with disturbances on the control

According to the simulation results obtained in FIL mode (Figure 14), good tracking and good stability are observed. The proposed controller shows good robustness despite the disturbance injected into the control signal at time t = 20s. The FMRLC controller manages to adapt to the reference changes and rejects the applied disturbance.

5.2. Experimental results

In this part, the experimental results obtained during the position control of the DC motor using the Zedboard FPGA are presented. In order to verify the efficiency of the proposed algorithm, the results obtained with the FMRLC controller were compared with those obtained with the PID controller.

5.2.1. PID controller

The PID controller parameters were tuned using the PID Tuner tool in Simulink. The parameters obtained are as follows: P = 21, I = 1.9, and D = 15. The experimental results obtained with the PID controller are shown in Figures 15 and 16.

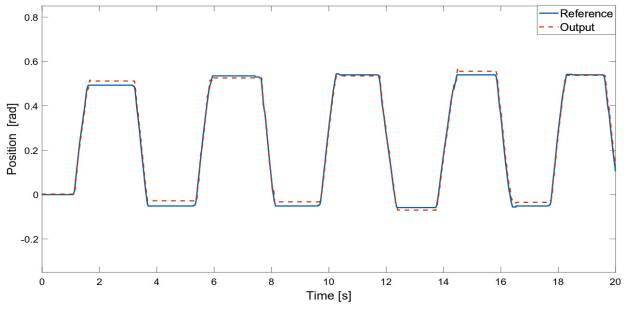

Figure 15. Experimental results of position tracking with PID controller

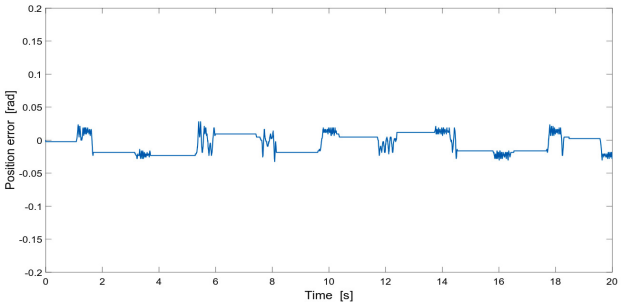

Figure 16. Position tracking error with PID controller

Figure 15 shows a test with a reference varying between 0.04 rad and 0.55 rad, generated by an encoder similar to that of the DC motor. As can be seen, the PID controller performs well, with a short response time that quickly converges to the desired reference signal. The tracking error is about ±0.03 rad, as shown in Figure 16.

5.2.2. FMRLC controller

Figures 17 and 18 show the experimental results obtained with the FMRLC controller.

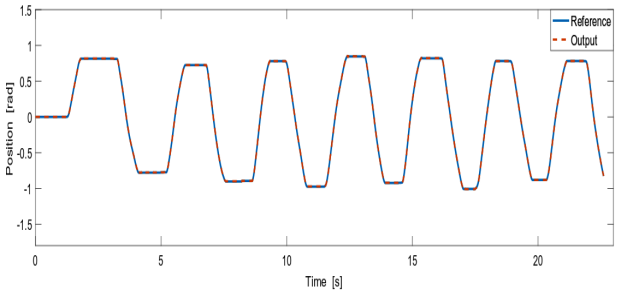

Figure 17. Experimental results of position tracking with FMRLC controller

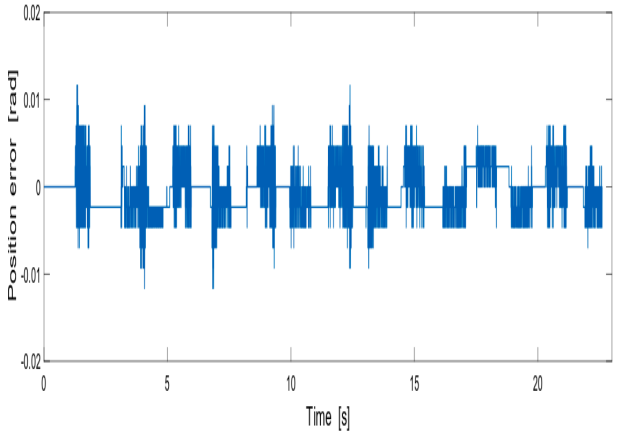

Figure 18. Position tracking error with FMRLC controller

The generated setpoint varies between −0.9 and 0.9. Excellent tracking can be seen here. The FMRLC controller closely tracks the reference signal with a very short response time, as shown in Figure 17. This is reflected in a tracking error of

about ±0.012. This error is practically inferior to that obtained with the PID controller, as shown in Figure 18.

The experimental results obtained show good position tracking and stability, with a lower error compared to the results of the PID controller. These results are explained on the one hand by the use of the knowledge base modifier, which modifies the membership functions in real time according to the DC motor dynamics, and on the other hand by the use of the Zedboard FPGA, which provides very short sampling times and parallel computation, accelerating the computation time to quickly converge to the desired performance.

6. Conclusion

This paper presents the implementation of a fuzzy logic controller FMRLC on a Zedboard FPGA to control the position of a DC motor. The idea was to implement a fuzzy controller with learning on an FPGA board. The use of this configurable circuit, which offers parallel data computation at a high operating frequency, aims at accelerating the computations, in particular the learning process, which requires a significant computation time to converge to the desired performance. The design and implementation of the controller were carried out using the MATLAB-Simulink environment. This methodology allows great design flexibility by using simple blocks on Simulink without resorting to low-level programming languages (VHDL/Verilog), thus reducing the design time of the algorithm. The fixed-point data type is used to optimize the use of hardware resources on the FPGA. This binary representation can affect the accuracy of calculations, but the design results in low power and resource consumption compared to floating-point based designs.[34]

The FMRLC controller was first tested using the FIL technique in Simulink, then implemented on the Zedboard and applied to the experimental device. The FIL simulation results showed good tracking performance and good robustness to disturbances. The experimental results showed the efficiency of the proposed controller compared to the conventional controller (PID), which can be explained by the controller adapting the parameters of the membership functions to each different situations. The design methodology used to implement the algorithm on the Zedboard FPGA board allowed to obtain a fast and accurate algorithm with an optimal consumption of hardware resources. The co-simulation results and the experimental results demonstrated the effectiveness of this design methodology.

Acknowledgments

This paper is derived from a research grant funded by the Research, Development, and Innovation Authority (RDIA), Kingdom of Saudi Arabia, with grant number 13382-psu-2023- PSNU-R-3-1-EI-. This research is supported by Automated Systems and Computing Lab (ASCL), Prince Sultan University, Riyadh, Saudi Arabia. The authors would like to thank Prince Sultan University, Riyadh, Saudi Arabia for supporting this work.

Fundings

This paper is funded by Prince Sultan University, Riyadh, Saudi Arabia. The authors would like to thank Prince Sultan University for paying the article processing fee for this paper.

Conflict of interest

The authors declare that have no conflict of interest.

Author contributions

Conceptualization: Mohand Achour Touat, Hocine Khati, Hand Talem, Ahmad Taher Azar
Formal analysis: Mohand Achour Touat, Arezki Fekik, Rabah Mellah, Ahmad Taher Azar, Saim Ahmed
Methodology: Mohand Achour Touat, Hocine Khati, Arezki Fekik, Ahmad Taher Azar
Writing – original draft: Mohand Achour Touat, Hocine Khati, Arezki Fekik, Ahmad Taher Azar, Saim Ahmed
Writing – review & editing: Arezki Fekik, Ahmad Taher Azar, Rabah Mellah, Saim Ahmed

Availability of data

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author/s.

References

1. Kemal U, Beyza Nur A. Adaptive MIMO fuzzy PID controller based on peak observer. *Int J Optim Control: Theor Appl.* 2023;13(2):139–150.
2. Masjudin, Alimuddin, Aisah SN, Wiryadinata R. Dc motor speed control based on fuzzy adaptive with fuzzy model reference learning control (fmrlc) algorithm. *In: 2020 2nd International Conference on Industrial Electrical and Electronics (ICIEE). IEEE*; 2020: 79-83.
3. Devashish J, Arifa A, Sanatan K, Debanjan R. Fuzzy-PID and interpolation: a novel synergetic approach to process control. *Int J Optim Control: Theor Appl.* 2024;14(4):355-364.

4. Fajrianto MR, Wahyudi W, Sudjadi S. Perancangan kontroler fuzzy model reference learning control (fmrlc) berbasis mikrokontroler atmega16 sebagai kendali motor brushless dc (bldc). *Transient: J Ilmiah Teknik Elektro*2017;6:597–604.
5. Saim A, Haoping W, Yang T. Fault tolerant control using fractional-order terminal sliding mode control for robotic manipulators. *Stud. Inform. Control.* 2018;27(1):55-64.
6. Saim A, Ahmad Taher A, Ibraheem KI. Nonlinear system controlled using novel adaptive fixed-time SMC. *AIMS Math* 2024;9(4):7895-7916.
7. Kopasakis G, Kopasakis G. Adaptive performance seeking control using fuzzy model reference learning control and positive gradient control. *In: 33rd Joint Propulsion Conference and Exhibit*; 1997: 3191.
8. Zhen L, Xu L. Fuzzy learning enhanced speed control of an indirect field-oriented induction machine drive. *IEEE Trans Control syst Technol.* 2000;8:270–278.
9. Mayhan P, Washington G. Fuzzy model reference learning control: a new control paradigm for smart structures. *Smart Mater struct.* 1998;7:874.
10. Layne JR, Passino KM, Yurkovich S. Fuzzy learning control for antiskid braking systems. *IEEE Trans Control Syst Technol.* 1993;1:122–129.
11. Reay D, Dunnigan M. Learning issues in model reference based fuzzy control. *IEEE Proc Control Theor Appl.*, 1997;144:605–611.
12. Duka AV, Oltean SE, Dulau M. Model reference adaptive control and fuzzy model reference learning control for the inverted pendulum. *comparative analysis. In: Proceedings of WSEAS International Conference on Dynamical Systems and Control*; 2005:168–173.
13. Monmasson E, Idkhajine L, Cirstea MN, Bahri I, Tisan A, Naouar MW. Fpgas in industrial control applications. *IEEE Trans Ind Informat* 2011;7:224-243.
14. Hace A, Franc M. Fpga implementation of sliding-mode control algorithm for scaled bilateral teleoperation. *IEEE Transac Ind Inform.* 2012;9:1291–1300.
15. Fekik A, Khati H, Azar AT, et al. FPGA in the loop implementation of the PUMA 560 robot based on backstepping control. *IET Control Theor Appl.* 2024;18(15): 1877-1891.
16. Huerta-Moro S, Tavizón-Aldama JD, Tlelo-Cuautle E. FPGA implementation of sliding mode control and proportional-integral-derivative controllers for a DC–DC buck converter. *Technol.* 2024;12(10):184.
17. Wang J, Li M, Jiang W, Huang Y, Lin R. A design of FPGA-based neural network PID controller for motion control system. *Sens.* 2022;22(3):889.
18. Li Y, Li SE, Jia X, Zeng S, Wang Y. FPGA accelerated model predictive control for autonomous driving. *J Intell Connect Vehicl.* 2022;5(2):63-71
19. Meghwal R, Yadav VK, Vardia M. Robust fuzzy controller design with FPGA implementation for matrix converter based induction motor drive. *e-Prime Adv Elect Eng Electron Energy* 2024;10:100752.
20. El Attafi A, El Alami H, Bossoufi B, et al. Robust control of a wind energy conversion system: FPGA real-time implementation. *Heliyon.* 2024; 10(15):e35712.
21. Chakravarty S. Technology and Engineering Applications of Simulink. *BoD–Books on Demand*; 2012.
22. Akkaya Ş, Üzgün HD, Akbati O. Fuzzy logic controller implementation with fpga in the loop simulation. *In: Proceedings of the 2017 International Conference on Mechatronics Systems and Control Engineering* ; 2017: 33-37.
23. Akbatı O, Üzgün HD, Akkaya S. Hardware-in-the-loop simulation and implementation of a fuzzy logic controller with fpga: case study of a magnetic levitation system. *Trans Instit Measure Control.*, 2019;41:2150-2159.
24. Deliparaschos K, Nenedakis F, Tzafestas SG. Design and implementation of a fast digital fuzzy logic controller using fpga technology. *J Intell Robot Syst.*, 2006;45:77-96.
25. Khati H, Mellah R, Talem H. Neuro-fuzzy control of a position-position teleoperation system using fpga, *In: 2019 24th International Conference on Methods and Models in Automation and Robotics (MMAR). IEEE*; 2019:64-69.
26. Khati H, Talem H, Mellah R, Bilek A. Neuro-fuzzy control of bilateral teleoperation system using fpga. *Iranian J Fuzzy Syst.* 2019;16:17-32.
27. Azzouz B, Hadjira B. Hardware/software codesign for intel ligent motor drive on an fpga, *In: 2020 2nd International Workshop on Human-Centric Smart Environments for Health and Well-being (IHSH). IEEE*; 2021; 227-232.
28. Lotfy A, Kaveh M, Mosavi M, Rahmati A. An enhanced fuzzy controller based on improved genetic algorithm for speed control of dc motors. *Analog Integr Circ Signal Process.* 2020;105:141-155.
29. Abdelkrim H, Othman SB, Saoud SB. Fpga implementation of self-reconfigurable fuzzy logic controller. *In: 2018 International Conference on Advanced Systems and Electric Technologies(IC ASET). IEEE*; 2018:151-156.
30. Moussa I, Khedher A. Real-time wte using flc implementation on fpga board: theoretical and experimental studies. *In: 2020 17th International Multi-Conference on Systems, Signals & Devices(SSD). IEEE*; 2020: 428-433.
31. Anand, M.S., Tyagi, B. (2012). Design and implementation of fuzzy controller on fpga. *Int J Intell Syst Applicat.* 2012; 10:35-42.
32. Layne JR, Passino KM. Fuzzy model reference learning control. *J Intell Fuzzy Syst.* 1996;4:33–47.
33. Ljung L. System Identification Tool-box: User's Guide. Citeseer; 1995.

34. Finnerty A, Ratigner H. Reduce power and cost by converting from floating point to fixed point. In: WP491 (v1. 0) ; 2017: 9-10.
35. Crockett LH, Elliot RA, Enderwitz MA. The Zynq Book Tutorials for Zybo and Zedboard. *Strathclyde Academic Media*; 2015.
36. Khati H, Fekik A, Azar AT, et al. Optimizing UAV stability and control with FPGA-based PID control system design. *In: 2024 International Conference on Control, Automation and Diagnosis (ICCAD). IEEE* ;2024:1-6.
37. Fekik A, Khati H, Azar AT, Hamida ML, Denoun H, & Kamal NA. FPGA-based performance evaluation of backstepping control and computed torque control for industrial robots. *Int J Automat Control.* 2025;19(1):101-132.

Mohand Achour Touat *was born in 1977. He earned his state engineer diploma in Automation in 2002 from Mouloud Mammeri University in Algeria. In 2005, he obtained a specialist diploma in Control Systems from the National Aviation University of Ukraine. He later received his PhD in Engineering Sciences, specializing in Computer-Aided Design. Currently, he is a lecturer in the Automation Department at Mouloud Mammeri University, where he has been the head of the department since 2016. His current research focuses on robuste control and artificial intelligence.*
https://orcid.org/0000-0002-5758-7653

Hocine Khati *obtained the Bachelor's and Master's degrees in Automation from Mouloud MAMMERI University of Tizi Ouzou (UMMTO), Algeria, in 2013 and 2015, respectively. In 2020, he obtained the Dotoral degree in Automatic Control from the same institution. He is currently an Associate Professor in the Automatic Department at UMMTO. His current research focuses on digital control, adaptive control, robotics, and artificial intelligence.*
https://orcid.org/0009-0004-5556-6308

Arezki Fekik *is a lecturer at Akli Mohand Oulhadj University in Bouira, Algeria. He currently works as a research engineer and is responsible for experimental testing and research in electrical systems control at the Laboratoire des Sciences Numériques de Nantes (LS2N), École Centrale de Nantes. He has published over 65 journal articles, conference papers, and book chapters in the fields of power electronics and its applications. His current research focuses on power electronics and its applications, such as wind turbines, photovoltaic systems, reliability, harmonics, microgrids, and variable speed drives.*
https://orcid.org/0000-0001-5053-1764

Ahmad Taher Azar *is a full Professor at College of Computer and Information Sciences (CCIS), Prince Sultan University, Riyadh, Saudi Arabia. He is a leader of Automated Systems and Computing Lab (ASCL), Prince Sultan University, Saudi Arabia. He is currently an editor for IEEE Transactions on Fuzzy Systems, IEEE Systems Journal, IEEE Transactions on Neural Networks and Learning Systems, Springer's Human-centric Computing and Information Sciences, and Elsevier's Engineering Applications of Artificial Intelligence. Prof. Azar was named one of the top 2% of scientists in the world in Artificial Intelligence by Stanford University, based on single-year impact and career-long impact. These rankings were published by Stanford University in the PLOS journal and were based on the SCOPUS database. Prof. Azar has expertise in Artificial Intelligence, Control Theory and Applications, Robotics, Machine Learning, Computational Intelligence and dynamical system modeling. He has authored/co-authored over 500 research papers in prestigious peer-reviewed journals, book chapters, and conference proceedings.*
https://orcid.org/0000-0002-7869-6373

Hand Talem *obtained his mechanical engineering degree in 2007 from the maritime school in Bousmail, Tipaza, Algeria, and his automatic engineering degree from the Université Mouloud MAMMERI de Tizi Ouzou (UMMTO), Algeria, in 2009. In 2010, he enrolled in the Magister en Automatique program at the same institution. He is currently the Chief Engineer at the Algerian shipping company CNAN. His current research interests include numerical control, adaptive control, robotics, and artificial intelligence.*
https://orcid.org/0009-0002-8483-7221

Rabah Mellah *received the Doctoral degrees in Automatic Control from USTHB University of Algeria in 2006. He is a professor at the Department of Automatic, UMMTO University, Tizi-Ouzou, Algeria. His research interests include nonlinear system, robotics, teleoperation, fuzzy logic and neural networks.*
https://orcid.org/0000-0003-2273-7532

Saim Ahmed *received his B.S degree in Electronics Engineering from Sir Syed University of Science and Technology, Karachi, Pakistan in 2009. He received his M.E degree in Industrial Control and Automation from Hamdard University, Karachi, Pakistan, in 2013. He completed his Ph.D. degree in control science and engineering from Nanjing University of Science and Technology, China in 2019. He has served as an Assistant Professor at the Department of Mechatronics, Shaheed Zulfikar Ali Bhutto Institute of Science and*

Technology, Karachi, Pakistan. He is currently working as a Postdoctoral Researcher at Prince Sultan University, Saudi Arabia. His research interests include the theory and applications of adaptive control, sliding mode control, time delay control, robotic exoskeleton and manipulators, nonlinearities and their compensation.

https://orcid.org/0000-0002-2302-705X

An International Journal of Optimization and Control: Theories & Applications
(https://accscience.com/journal/ijocta)