Journal of Information Systems Engineering and Management
2025, 10(28s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Digital System Identification Controller for Adaptive Feedback Control in Closed-Loop FES

M.K.Safie[1,2], E.Noorsal[1*], Z.Hussain[1], S.Z.Yahaya[1], S.S.M. Sallah[1], A.N.A Rashid[1], S.Arof[1,3]
[1] *Electrical Engineering Studies, College of Engineering, Universiti Teknologi MARA Cawangan Pulau Pinang, Permatang Pauh Campus, Pulau Pinang, Malaysia.*
[2] *Altera Semiconductor Technology, Plot 6, Medan Bayan Lepas, Bayan Lepas Technoplex, Bayan Lepas, Pulau Pinang, Malaysia*
[3] *Universiti Kuala Lumpur, Malaysian Spanish Institute Kulim Hi-Tech Park, Kulim, Kedah. Malaysia*
*Email: *emilia.noorsal@uitm.edu.my*

ARTICLE INFO

Received: 18 Dec 2024

Revised: 10 Feb 2025

Accepted: 28 Feb 2025

ABSTRACT

Functional Electrical Stimulation (FES) devices are widely used for spinal cord injury patients but often face challenges with nonlinearity effects, leading to premature muscle fatigue due to feedback controller discrepancies. Therefore, this research proposes a digital system identification controller (SIC) to determine the patient's condition by extracting important information from the patient's knee trajectory response, which includes time delay, rise time, overshoot, steady-state time, steady-state value, and steady-state error. The extracted information is beneficial in enabling real-time self-tuning of the feedback control system based on the patient's condition. Initially, the digital SIC was modelled using MATLAB Simulink, then hardcoded into digital logic using hardware description language (HDL) Verilog codes and verified using Quartus Modelsim. Thereafter, the performance of the digital SIC was tested with a knee extension model using the HDL co-simulation technique in MATLAB Simulink. The results showed that the digital SIC designed in HDL produces accurate output reading as MATLAB Simulink with error percentages ranging from 0% to 33.3%. The adaptive feedback controller could use this key data information to fine-tune the internal feedback settings. This evaluation demonstrates the controller's functionality and efficiency, providing insights into its suitability for adaptive feedback control applications.

Keywords: Adaptive Feedback Control (AFC), System Identification Controller (SIC), Functional Electrical Stimulator (FES), Verilog Hardware Description Language, Register-Transfer Level (RTL), Field Programmable Gate Array (FPGA)

INTRODUCTION

A permanent injury, spinal cord injury (SCI), can cause paralysis, pressure sores, muscle spasms, weakened muscles, and heart-related issues, affecting daily activities and quality of life [1]. It affects the spinal cord's nerves, resulting in motor deficits, including paralysis and caused by trauma like car accidents, sports injuries, or violence, resulting in limb movement loss in individuals with SCI [1, 2]. Spinal cord injuries can also arise from nontraumatic reasons, such as illnesses and tumours [2]. [1]. FES-assisted devices play a vital role in healthcare by aiding SCI patients in various rehabilitation exercises, including gripping objects, extending the elbows and knees, standing, walking, cycling, rowing, and sitting down [3, 4]. Functional electrical stimulation (FES) is used in rehabilitation to treat limb contractions [5]. During rehabilitation, FES uses external electrical pulses to stimulate muscles, adjusting their strength or weakness, similar to giving a push to help them perform tasks [6, 7]. The goal is to maintain and repair muscular function. However, due to factors including muscle exhaustion and spasms, people with spinal cord injuries (SCI) respond to FES in complex ways. The dynamic character of the muscle reaction must be taken into account by FES control system designers to overcome these difficulties. Strong control of the FES is essential since it interacts with the body's muscles and bones, which can cause noise and ambiguity [8].

A significant limitation of current FES devices lies in their open-loop controller designs. While an open-loop pulse can generate movement, it lacks the capability to evaluate or correct deviations from the intended motion [9]. This inability to update model parameters and control configurations in real-time can result in increased steady-state errors and diminished system performance [6]. The substantial nonlinearities, intrinsic time-variance, time delays, and current fluctuations of the neuromuscular skeletal system further render open-loop FES controllers unsuitable for precise movement control [9].

Closed-loop FES systems, though designed to address these issues, also face challenges due to the limitations of existing feedback controllers. Many contemporary feedback control algorithms are inadequate for real-world FES applications, as they fail to effectively manage nonlinear phenomena such as response delays, muscle fatigue, and spasticity. Advanced algorithms are required to enable accurate real-time adjustments of electrical stimulation pulses[1].

Various control strategies have been investigated to overcome these challenges, including Proportional-Integral-Derivative (PID) controllers [10], Fuzzy Logic [3, 4], Neuro-Fuzzy systems, Sliding Mode Controllers (SMC) [4], Adaptive Neural Networks (ANN), and Adaptive Neuro-Fuzzy Inference Systems (ANFIS). However, these controllers often struggle to effectively manage electrically stimulated muscle contractions and typically demand significant time for parameter tuning [3]. SMC, a widely used method in industrial systems for handling nonlinear systems with matched disturbances, has limitations, such as chattering and uncertainty sensitivity [11]. Chattering in sliding mode control can be mitigated through techniques such as the integral timing algorithm with disturbance estimation compensation, which improves dynamic performance but may not be ideal for nominal robotic systems [12].

Functional Electrical Stimulation (FES) feedback control algorithms are currently unsuitable for practical applications due to their inability to adapt to the nonlinear behavior of muscle function. Adaptive control techniques have been proposed as a potential solution to these limitations, but their effectiveness in addressing the complexities of specific nonlinearities remains underexplored. System identification controllers, which can determine and model system dynamics from input-output data, represent a promising approach in feedback control systems [4, 13]. Digital control offers two notable advantages: adaptive control and online programmability. By shaping the feedback signal, it enables communication between the converter and processing unit, facilitating real-time estimation of closed-loop system characteristics without the need for external testing equipment [14].

Since open-loop controllers are inadequate for managing voluntary upper-body movements, muscle fatigue, or spasticity, feedback control principles are applied to generate functional movements. To address the complexities of Functional Electrical Stimulation (FES) systems, nonlinear control strategies are recommended [16]. Conventional control systems can handle minor deviations effectively but struggle with uncertainties and changes in controller configuration. To overcome these limitations, adaptive control with model identification (MIAC) offers a solution by dynamically adjusting optimal controller settings based on system input-output data [17].

This study investigates an adaptive feedback control strategy for complex nonlinear systems, tackling both known and unknown nonlinearities as well as external disturbances [15]. Specifically, a system identification controller is proposed within the feedback control unit to monitor and facilitate knee extension movements, reducing muscle fatigue and enabling prolonged rehabilitation exercises.

Field programmable gate arrays (FPGAs) and application-specific integrated circuits (ASICs) are critical in implementing advanced control systems due to their speed, concurrency, real-time processing, and parallel computing capabilities [3]. While ASICs offer low power consumption and hardwired reliability, their lengthy design cycles and high manufacturing costs make them less cost-effective and inflexible for reprogramming [18]. In contrast, FPGAs have gained popularity due to advancements in VLSI technology, offering advantages such as programmable hardwired features, shorter design cycles, fast time-to-market, CPU inclusion, low power consumption, and higher density [19].

This project aims to improve closed-loop FES performance through the development of a system identification controller capable of real-time self-tuning and adapting to patient-specific conditions. The main objectives of the project include designing a digital system identification controller for adaptive feedback control within closed-loop FES. Additionally, the project will involve implementing system identification using Hardware Description Language (HDL). Finally, the system will be simulated and verified using an HDL co-simulation tool, such as MATLAB Simulink. The digital system identification controller for AFC in closed-loop FES will extract information from knee response and be implemented using a digital VLSI flow. The design will be simulated and verified using HDL co-simulation tools and rigorously analyzed for performance.

METHODOLOGY

This section explains the system focuses on an Adaptive Feedback Controller (AFC) designed to regulate knee response, starting with an overview of its core components. It includes an extension of the knee model, allowing for more accurate control. The extraction of knee response is carried out using the three-point and eight-point moving average methods to ensure smooth data processing. The architecture of the Adaptive Feedback Controller is then explained, followed by a detailed breakdown of the system identification controller for knee response, covering its internal structure and finite state diagram. To demonstrate practical implementation, the system is simulated at the system level in MATLAB Simulink. Finally, a flowchart is provided to visually represent the system's processes and interactions.

2.1. System Overview of Adaptive Feedback Controller (AFC)

Figure 1 provides an overview of the Adaptive Feedback Controller (AFC). The AFC is a critical tool for ensuring precise knee extension to a specific angle during rehabilitation activities. It demonstrates robustness against perturbations and accounts for nonlinear effects. By comparing the current knee angle with the target angle, the controller determines the appropriate control output signal. Through both closed-loop and open-loop simulations, the AFC captures key system characteristics and adjusts its parameters to deliver optimal treatment outcomes for patients.

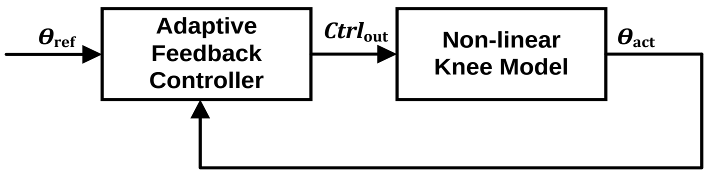

Figure 1. Overview of an adaptive feedback controller (AFC) [4].

2.2. Knee Extension Model

Figure 2 illustrates the estimated knee movement in a spinal cord injury patient, represented as a knee angle. This study employs a closed-loop FES device to induce muscle contractions and facilitate knee extension in such patients. The knee angle is initially set to 0, and an adaptive feedback controller adjusts the stimulus pulse width to achieve the desired angles. The error is defined as the difference between the desired and actual angles. The error and change in error are precisely defined by equations (1) and (2).

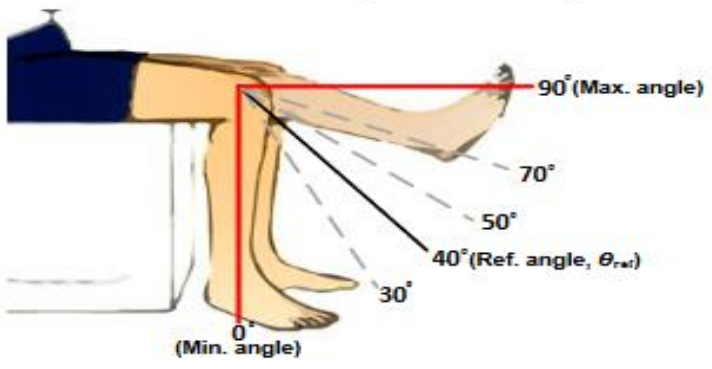

Figure 2. Example of knee extension movement [4].

These input data are utilized by the adaptive feedback controller to calculate the appropriate charge required to achieve the desired angle.

$$\textit{Error, } e = \textit{reference angle} - \textit{actual angle} \quad (1)$$

$$\textit{Change in error, } de = \textit{current error} - \textit{previous error} \quad (2)$$

2.3. Extraction of Knee Response using System Identification Controller

System identification was used to extract essential features (time delay, rise time, overshoot, and steady-state error) from the acquired knee response. The following subsections explain the details and illustrations of each feature extraction operation.

2.3.1. Example of Time Delay Extraction

Figure 3 illustrates the process of extracting the time delay from the knee angle trajectory using a three-point moving average technique. The acquired data are stored in memory units (m1, m2, and m3) with a specified time interval between them.

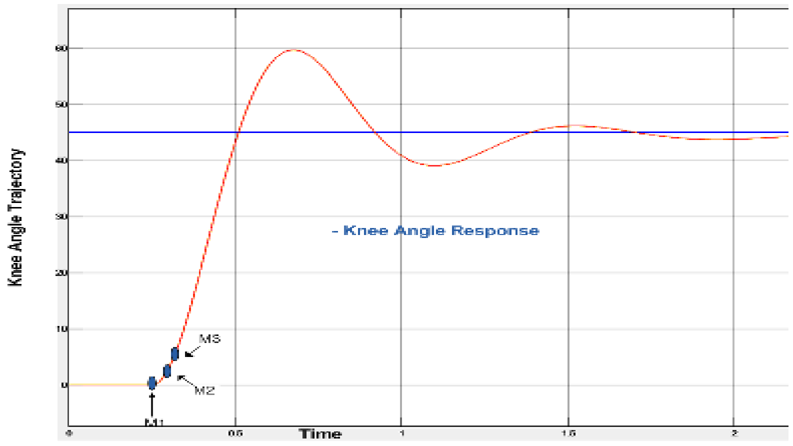

Figure 3. An example of time delay extraction using a three-point (m1, m2, m3) moving average method [4].

An internal counter increments at specified intervals, and the time delay is calculated using Equation (3) by multiplying the counter's value by the time interval and subtracting the acquired interval [4].

$$\text{Time Delay} = \text{Counted Value} \times \text{Time Interval} - (2 \times \text{Time Interval}) \tag{3}$$

2.3.2. Example of Rise Time Extraction

Figure 4 illustrates the FES open-loop system, which measures knee angles. The counting process begins when 10% of the target angle is reached and stops at 90% of the target angle. The rise time is then determined by multiplying the counted value by the time interval.

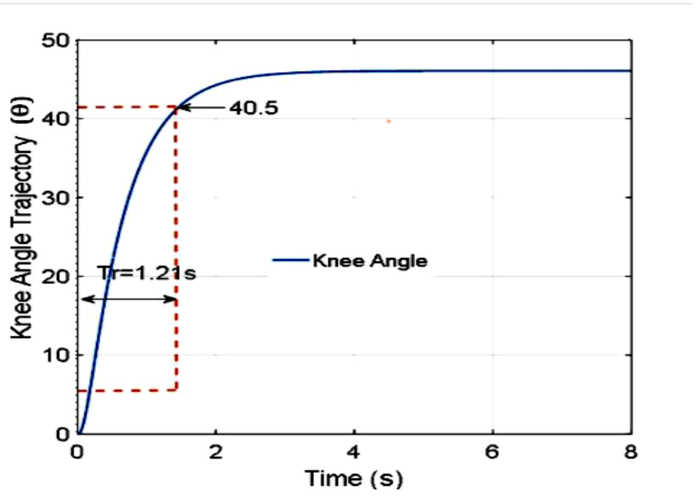

Figure 4. An example of rise time extraction using a three-point (m1, m2, m3) moving average method.

2.3.3. *Example of Overshoot Extraction*

Figure 5 illustrates the calculation of the overshoot value, determined by subtracting the steady-state value from the maximum peak value using the three-point moving average method. This value was utilized to determine the appropriate sliding mode (SM) gain setting for closed-loop operation. [4].

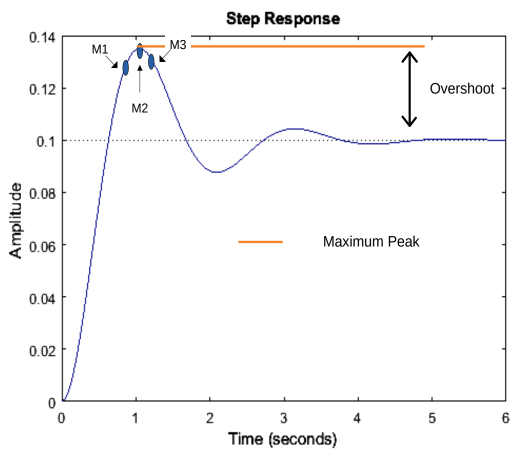

Figure 5. An example of overshoot extraction using a three-point (m1, m2, m3) moving average method.

$$Overshoot = Maximum\ Peak - Steady\ State\ Value \quad (4)$$

2.3.4. *Example of Steady-State Value Extraction*

Figure 6 illustrates the extraction of the steady-state value using the eight-point moving average method, involving memory units m-4, m-3, m-2, m-1, m0, m1, m2, and m3 with specified time intervals. The steady-state value was identified when all memory unit values were consistent.

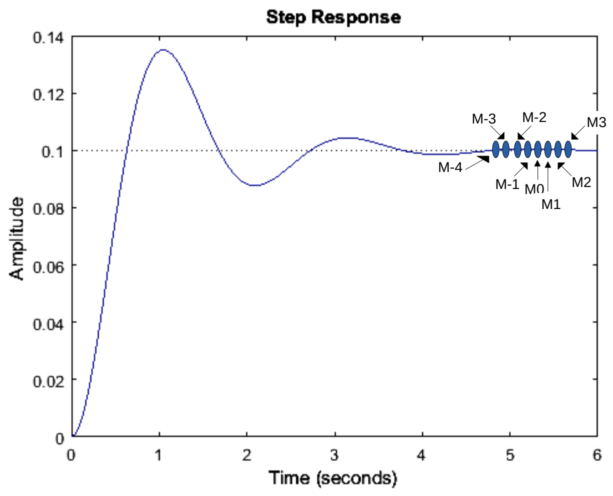

Figure 6. An example of steady-state value extraction using eight-point (m-4, m-3, m-2, m-1, m0, m1, m2, m3) moving average method.

The steady-state error was calculated using Equation 5 [4].

$$Steady\ State\ Error = Reference\ Angle - Steady\ State\ Value \quad (5)$$

The steady-state time was determined by multiplying the value counted by the internal counter with the counter's delay time interval, as shown in Equation 6.

$$\textit{Steady State Time} = \textit{Counted Values} \times \textit{Time Interval} \quad (6)$$

2.3. System Identification Algorithm for Knee Response

Table 1 illustrates the algorithm for the system identification controller. These algorithms show an overview to extract and calculate the features needed in knee response sample data. The extracted features (time delay, rise time, overshoot, steady state value, steady state time and steady-state error) from the system identification process will be used to determine the patients' status under treatment and select appropriate control laws, gain settings, and reference angles. A three-point moving average method consisting of three memory units (m1, m2, and m3) and internal counters were used to extract the essential features (time delay, rise time, overshoot, steady state value, steady state time and steady-state error), from the knee response. The algorithm starts by calculating the time delay at every 10ms in which the acquired knee response data are stored into eight memory units by using three-point moving average method. Then the rise time is calculated by activating the internal counter when the knee angle response reaches 10% and stops counting when the knee angle reaches 90%.

To determine the steady-state value, error, and time in the knee response analysis, the system first ensures that both the time delay and rise time have been calculated. Additionally, the counter must have exceeded 100 counts, indicating that the system has had sufficient time to stabilize. The next step involves checking the knee angle values stored in the eight memory units (m-4, m-3, m-2, m-1, m0, m1, m2, and m3). A moving average is applied to these values, and if all eight points remain constant, the system is considered to have reached a steady state. The steady-state value is then determined from these stable readings. Once the steady-state value is identified, the steady-state error is calculated by subtracting this value from the reference angle (which is 45° in this case). This error helps evaluate how closely the system maintains the desired knee position. Then the steady-state time is computed using the counter value at the point of stabilization. This value is multiplied by the system's internal time delay (0.01 seconds per count), giving the total time required to reach the steady state. Finally, to calculate overshoot in the knee response, the system first identifies the maximum peak using a three-point moving average of recent data points (m1, m2, and m3). A peak is detected if m2 is greater than both m1 and m3, meaning that the knee angle has reached a local maximum. When this condition is met, the system stores the value of m2 in a register called max_reg, which temporarily holds the highest recorded peak. Next, the system compares max_reg with the latest m3 value to ensure it captures the highest peak. If max_reg remains greater than m3, it is confirmed as the maximum peak. Once the peak is identified, overshoot is calculated by subtracting the steady-state value from this peak. This method helps evaluate how much the knee angle exceeds its target before stabilizing, which is essential for assessing system performance and tuning the FES response for better control.

Table 1. System Identification Controller Algorithm

System Identification Algorithm (During Testing Phase)	
1	**#1. Calculate the time delay.**
2	The internal counter for time delay is activate when FES System generate the sample data.
3	The sample data acquired every 10 ms from knee response and store in eight memory units (m-4, m-3, m-2, m-1, m0, m1, m2, and m3)
4	Use the three-point (m1, m2, and m3) moving average and counter to calculate the time delay.
5	Calculate the time delay when m3>m2 and m1 = 0
6	**#2. Calculate the rise time**
7	Use a counter that starts counting when the knee angle reaches 10% and stops counting when it reaches 90% to determine the rise time if the obtained knee angle is between 10% and 90% of the goal angle (45°).
	#3. Calculate the steady-state value, error, and time
8	If time delay and rise time have been calculated and the counter value > 100
9	Check eight points (m-4, m-3, m-2, m-1, m0, m1, m2, and m3) moving average
10	If the difference between m-4, m-3, m-2, m-1, m0, m1, m2, and m3 are 0 then calculate the steady state value
11	Then calculate the steady-state error: Reference angle – steady state value angle
12	Calculate steady state time: Counted value x internal time delay (0.01 s)

```
#4. Calculate overshoot
Find the maximum peak using moving averages of three points (m1, 2, mand m3)
If m2 > m1 and m2 > m3, then store the peak value in the max_reg
Compare the max_reg value with m3 value to get the highest peak value
If max_reg > m3, Maximum peak is max_reg
Calculate overshoot: Maximum peak (max_reg) – steady state value
```

2.4. System Identification Controller

Figure 7 depicts an overview of adaptive feedback controller, which includes the System Identification Controller (SIC), the Main Control Unit (MCU), and a Feedback Controller Unit. The system identification controller extracts key features such as rise time, time delay, steady-state error, steady-state time, steady-state value, and overshoot from the acquired knee trajectory output response for testing purposes.

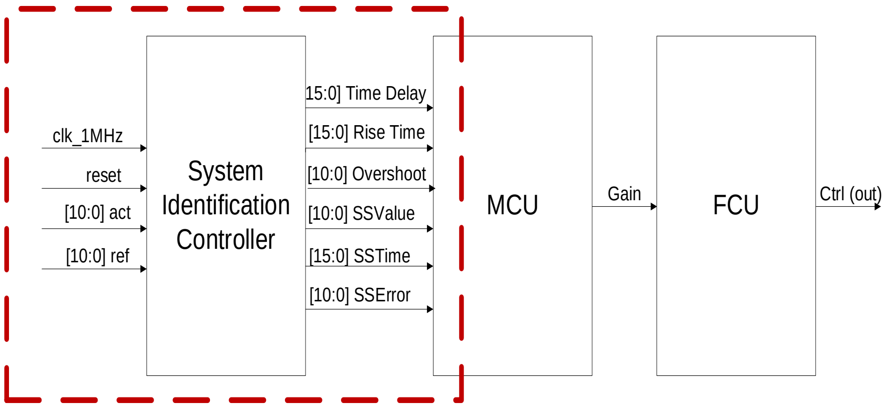

Figure 7. Adaptive feedback controller with system identification.

The SIC was designed using hardware description language (HDL) Verilog code. It extracts essential features from the knee response to provide accurate information to the feedback control unit. The extracted features are utilized by the feedback control unit to optimize the gain setting, reference angle, and control laws. The three-point and eight-point moving average methods are employed to extract these features accurately.

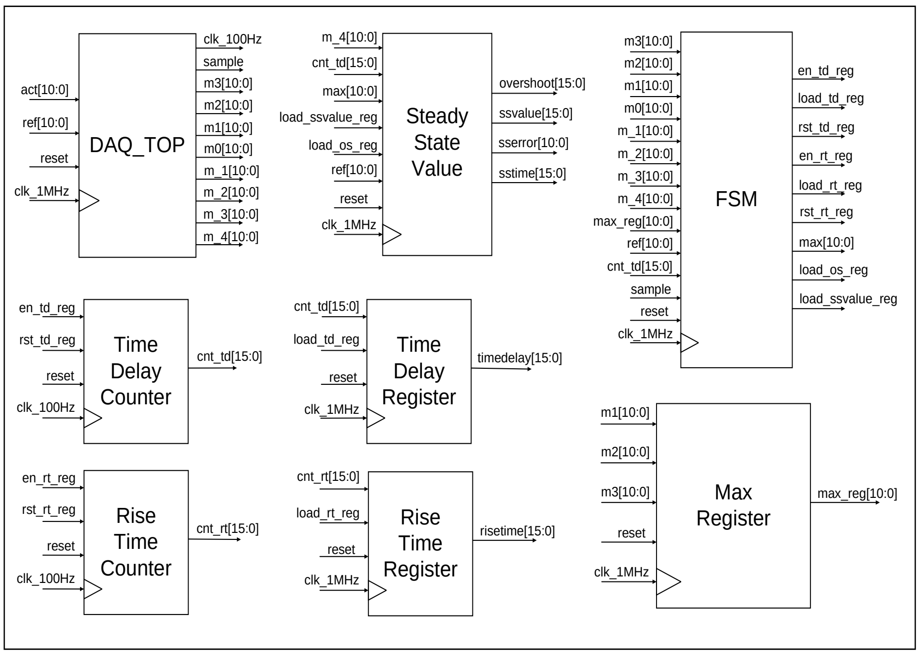

Figure 8. Internal architecture of system identification controller.

The internal architecture of the SIC used in this project is shown in Figure 8. The SIC consists of eight fundamental submodules. This system generates a sample signal every 10 milliseconds to acquire and store knee data in the memory units within the DAQ_Top submodule. The Time Delay Counter and Rise Time Counter operate using the "100Hz" clock. The data in the DAQ block of the DAQ_Top module continuously moves through the memory units, transitioning from real-time data to the memory units (from m3 to m-4). There are a total of eight memory units in the DAQ section.

The Time Delay Counter uses the "100Hz" clock to generate a counted value every 10 ms and will automatically start operating when the "en_timedelay_reg" signal is set to 1. Otherwise, the counter remains inactive. The counted value is saved to the Time Delay Register when the "load_timedelay_reg" signal from the FSM module is 1. Similarly, the Rise Time Counter also uses the "100Hz" clock to produce a counted value every 10 ms. When the first memory unit, "m3," exceeds 10% of the target angle, the FSM module sets the "en_risetime_reg" signal to 1, initiating the counter. When "m3" reaches 90% of the target angle, the "reset_risetime_reg" signal is set to 1, stopping the counter. The Rise Time Register generates the rise time value when the "load_risetime_reg" signal is 1.

For the Max Register, three memory units, "m1," "m2," and "m3," serve as inputs. The data in these memory units are compared to determine the maximum value among them. In the Steady-State Value module, the steady-state value, steady-state time, steady-state error, and overshoot are calculated once all steady-state conditions are met. When the steady-state value condition is satisfied, the "load_ssvalue_reg" and "load_overshoot_reg" signals are set to 1, initiating the counting of all values in the Steady-State module.

Figure 9 illustrates how the system identification controller operates using a state machine. Five key states are involved in this process: Idle, Time_Delay, Rise_Time, Overshoot, and Steady_State_Value. The procedure is initialized when the data processing is complete. In the Idle state, the FSM of the system identification controller begins operation. From the Idle state FSM transitions to a new state (Time_Delay) if the "sample," signal is activated to logic HIGH; otherwise, it remains in Idle state until the condition is met. The “sample” signal is generated every ten milliseconds to acquire the knee angle data and stored in memory units. In the Time_Delay state, the time delay duration is computed when the m3 data exceeds the m2 data, and the m1 data equals 0. The FSM then transitions to the Rise_Time state. During the Rise_Time state, the m3 data is analyzed.

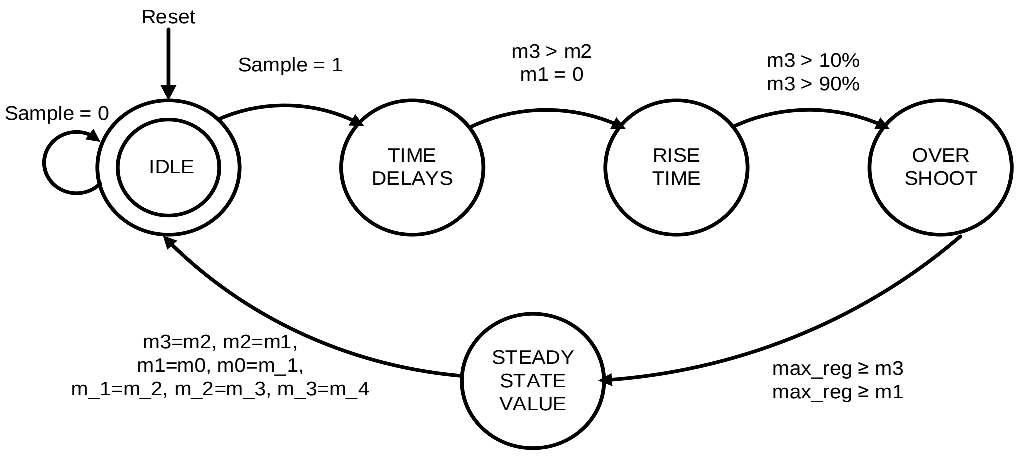

Figure 9. Finite state diagram

Once the conditions are met, the FSM transitions to Overshoot state. In this state, the data in m3 is compared to the maximum register value. If the data meets the criteria (max_reg >m3 && max_reg > m1), the FSM moves to the next state; otherwise, it continues comparing the values until the condition is satisfied. In the final state, known as the Steady-State_Value state, additional memory units (m-4, m-3, m-2, m-1) are examined and compared. All the data stored in the 8 memory units should be almost equal to ensure steady-state condition is achieved. This method ensures the system's steady-state value is accurate, as relying on only a small number of memory units would lead to imprecision.

2.5. System Level of System Identification Controller in MATLAB Simulink

The knee extension model used as the plant in the MATLAB Simulink HDL co-simulation and System Identification Controller (SIC), is shown in Figure 10. The SIC was initially developed using MATLAB functions and programming before being implemented in digital Hardware Description Language (HDL). To facilitate a comprehensive comparison, this simulation integrates both HDL co-simulation and MATLAB-based SIC, enabling the evaluation of both implementations within a unified framework.

The top section of the diagram, enclosed in a blue-dashed box, represents the MATLAB-based system. In this setup, the knee joint and muscle system are modelled, receiving a reference input angle and generating the corresponding actual knee angle as output. The system's performance is evaluated through key metrics, including time delay, overshoot, rise time, and steady-state value, which are visually displayed. In contrast, the bottom section, enclosed in an orange-dashed box, represents the HDL co-simulation-based implementation of the SIC. This approach bridges the gap between software-based validation and real-time hardware execution, ensuring that the control algorithm can be efficiently deployed in a hardware environment.

The efficiency of both the MATLAB and HDL SIC implementations was assessed using this co-simulation approach. The system response was analyzed for a target reference angle of 45°, with both the reference and actual angles scaled by a factor of 10 to improve the precision of digital data acquisition. The data was sampled at 10-millisecond intervals, allowing for accurate performance evaluation and comparison between the two implementations. This section includes additional digital processing blocks and conversion elements, ensuring compatibility between software models and hardware logic. Dedicated buttons for compiling and launching the HDL simulator indicate a workflow aligned with FPGA or ASIC-based verification. Overall, this setup highlights a systematic approach to control system validation, transitioning from theoretical modeling to hardware-optimized deployment.

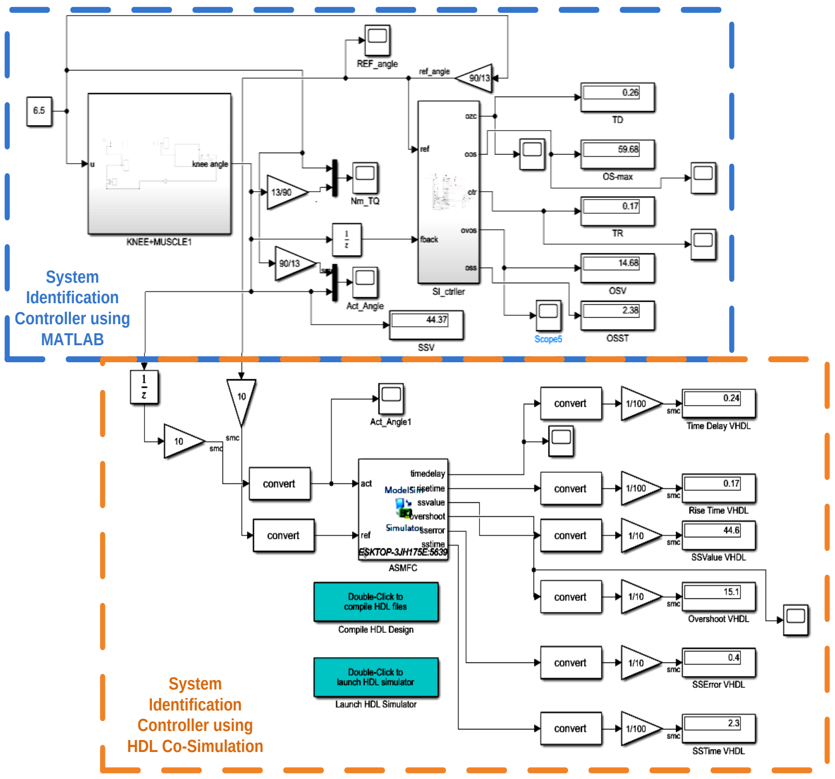

Figure 10. System Identification Controller Modelling using Knee Extension Model as Plant in MATLAB Simulink.

2.6. Design Process Flow

The flowchart outlines a systematic approach to designing and verifying a system identification controller using HDL Verilog and MATLAB Simulink as shown in Figure 11. The process begins with defining the system specifications, ensuring a clear understanding of the requirements. In the first stage, the researcher develops the internal architecture of the controller using HDL Verilog. Once the architecture is defined, testbenches are implemented to validate the functionality of the system identification submodules in Quartus. After testing, the design is evaluated against the specifications. If it does not meet the required performance, modifications and re-evaluations are performed until the criteria are satisfied. Once the design is validated, next stage begins, where the HDL model is imported into MATLAB Simulink for co-simulation. This step enables an in-depth analysis of the controller's output response to ensure it operates correctly under different conditions. After successfully verifying the system through simulation, the design process is concluded. This structured approach ensures accuracy, reliability, and efficiency in developing the system identification controller, making it ready for real-world implementation.

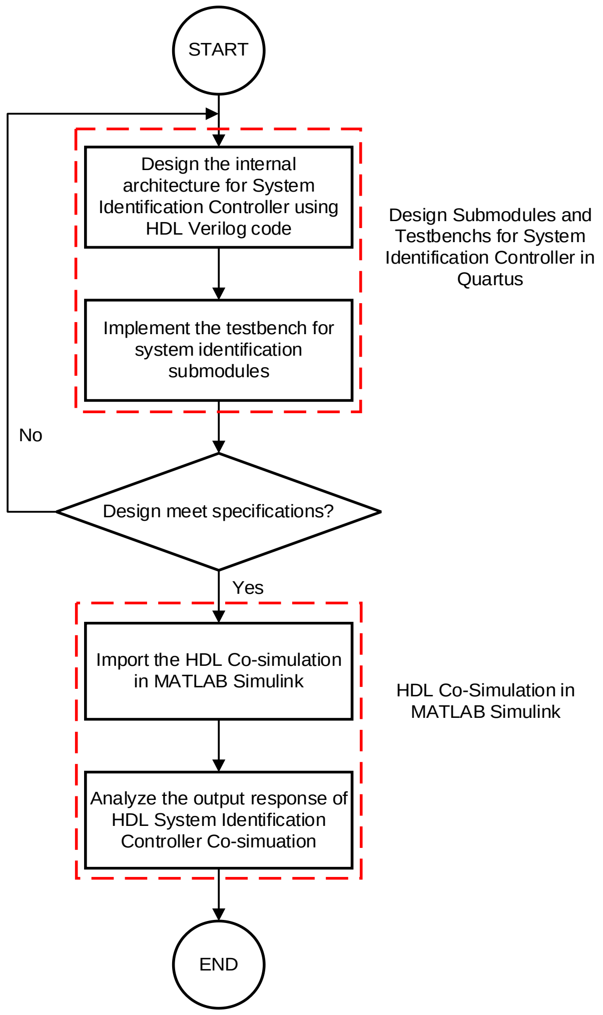
START
Design the internal architecture for System Identification Controller using HDL Verilog code
Implement the testbench for system identification submodules
Design Submodules and Testbenchs for System Identification Controller in Quartus
No
Design meet specifications?
Yes
Import the HDL Co-simulation in MATLAB Simulink
Analyze the output response of HDL System Identification Controller Co-simuation
HDL Co-Simulation in MATLAB Simulink
END

Figure 11. Flowchart of the system identification controller.

SIMULATION RESULT AND DISCUSSION

This section outlines the process of developing and testing digital SIC for knee response. It starts with the collection of knee response data from MATLAB. Using this data, the digital system identification controller is synthesized, incorporating the internal architecture of the SIC and relevant FPGA details. The system then undergoes RTL simulation, generating waveforms that are analyzed to evaluate its performance. Finally, the output responses from the HDL implementation and MATLAB Simulink are compared to assess the accuracy and consistency of the system identification across both platforms.

3.1. MATLAB Knee Response

Based on Figure 12, the knee response characteristics—time delay, rise time, overshoot, and steady-state value—can be determined. The calculation of time delay starts when the angle data is at 0. The rise time calculation begins when the data exceeds 10% of the reference angle and continues until it reaches 90% of the reference angle. Subsequently, the system reaches the peak value of the data before decreasing to the steady-state stage.

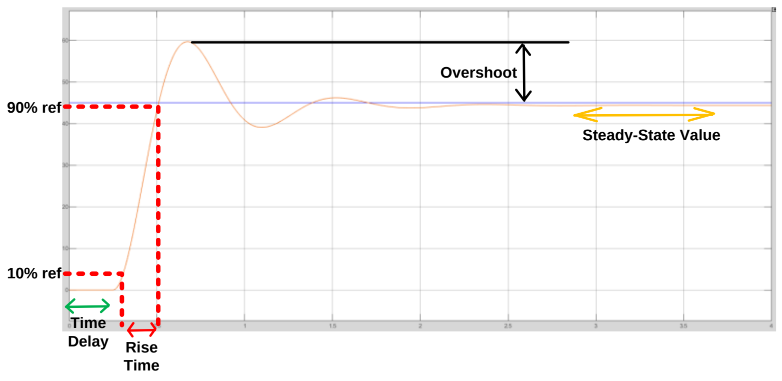

Figure 12. Knee response in MATLAB

According to Table 2, the MATLAB results show the value of all the features in knee responses. The features of knee response include the time delay, rise time, overshoot, steady-state value, and steady-state time. The recorded data are compared with HDL simulation results in order to determine the percentage inaccuracy between the two readings.

Table 2. The value of System Identification Controller Output Response in HDL Simulation

The features in Knee Response	MATLAB SIC
Time Delay	0.26s
Rise Time	0.17s
Overshoot	14.68°
Steady-State Value	44.37°
Steady-State Time	2.38s
Steady-State Error	0.6°

3.2. Synthesize Digital System Identification Controller

The internal architecture of the system identification controller of the AFC is shown in Figure 13 and is made up of eight submodules: DAQ_Top, Max Register, Steady-State Value, Time Delay Counter, Time Delay Register, Rise Time Counter, and Rise Time Register. Initially, the DAQ_Top line's "clk_1MHz" inputs are used as the inputs for every submodule with the exception of the rise time counter and time delay counter. Every submodule will use the "reset" as its input. The time delay register module will provide the output for the time delay, and the rise time register module will produce the output for the rise time value. At the steady-state value register, additional outputs will be produced, including steady-state value, steady-state time, steady-state error, and overshoot.

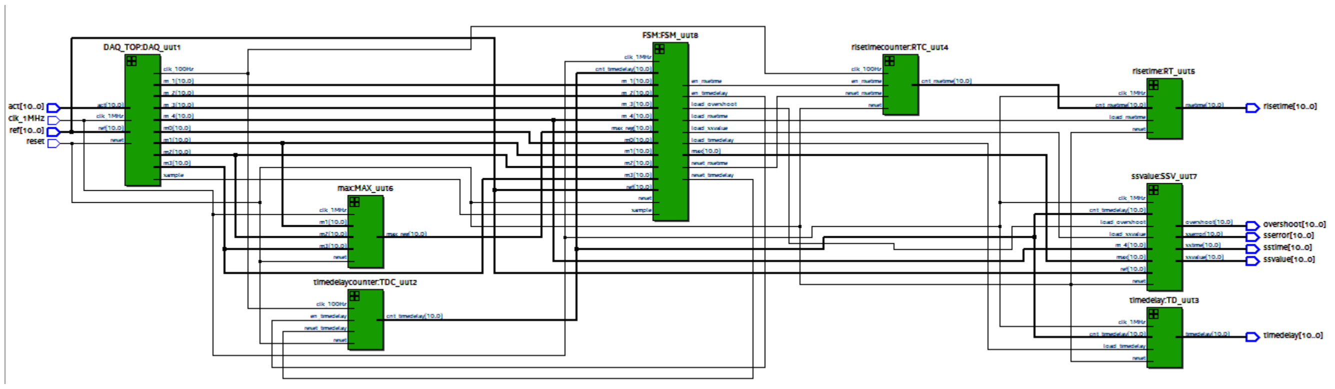

Figure 13. Internal Architecture of System Identification Controller.

Table 3 shows the synthesis summary report for the system identification controller using an Intel Field-Programmable Gate Array (FPGA). The total number of logic elements used for this digital system identification controller design was 334 (< 1% utilisation), the total number of registers was 262, and the total number of memory bits was 0 (0% utilisation), according to the synthesised results obtained. At a low temperature of 0 °C, the system could function at a maximum frequency (Fmax) of 187.34 MHz.

Table 3. Synthesize Summary Report for SIC using Intel Field-Programmable Gate Array (FPGA) (Cyclone IV E) Chip.

Items	**Types and Utilization**
Family name	Cyclone IV E
Device	EP4CE115F29C7
Total Logic Elements	355 / 114,480 (< 1 %)
Total Registers	287
Total pins	105 / 529 (20 %)
Total Memory bits	0 / 3,981,312 (0 %)
Embedded Multiple 9-bit elements	0 / 532 (0 %)
Total PLLs	0 / 4 (0 %)

3.3. RTL Simulation of Digital System Identification Controller

As indicated in Table 4, the actual value of time delay is calculated by multiplying the counted value by the time interval, which is 10 ms. This results in an actual time delay represented in standard seconds as 0.24 s. Similarly, for the rise time value, a counted value of 17 is generated by the 100 Hz counter. To obtain the precise rise time of 0.17 s, this counted value is multiplied by the 10 ms time interval.

Figure 14 displays the output of the SIC in Quartus using HDL Verilog code. According to Figure 14, the value of **A** is 24, which is the counted time delay using 100Hz clock frequency (10 ms clock period). In the HDL simulation, the actual data produced from the acquired knee response is multiplied by 10 to achieve accurate readings. Since the data was initially scaled by a factor of 10 for precision, the steady-state value (**C**) must be divided by 10. The same adjustment applies to the steady-state error (**E**) and overshoot (**D**). For the steady-state time (**F**), the counted value must be multiplied by 10 ms, as it is derived from the time delay counter.

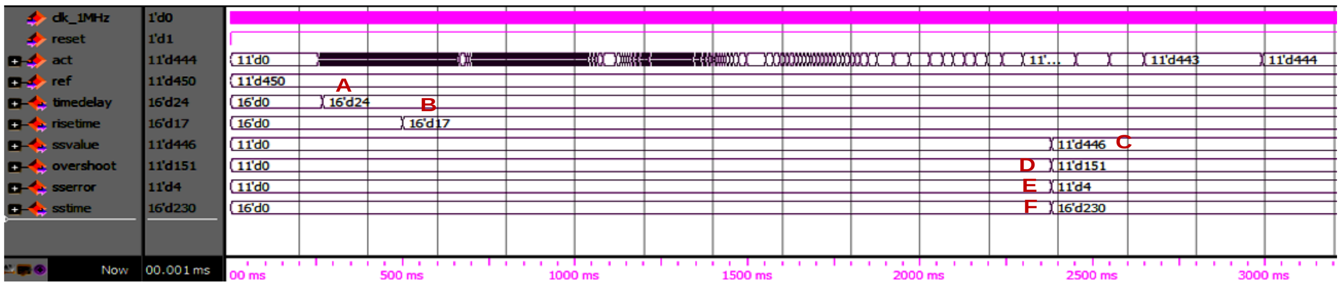

Figure 14. System Identification Controller Output

Table 4. The SIC Output Response in HDL Simulation

Parameter	**System Identification Controller of Knee Response**	
	HDL Simulation	***Actual Value***
Time Delay	24	0.24s
Rise Time	17	0.17s
Overshoot	151	15.1°
Steady-State Value	446	44.6°
Steady-State Error	23°	2.3°
Steady-State Time	4	0.4s

3.4. Comparison Output Response for System Identification between HDL and MATLAB Simulink

Table 5 compares the MATLAB-based System Identification Controller (SIC) and its HDL co-simulation counterpart in controlling a knee system. It indicates a slight difference in the percentage error between the HDL SIC and MATLAB SIC. The analysis reveals that the time delay in HDL SIC is slightly lower compared to MATLAB SIC, indicating a 7.69% improvement in response time. The rise time remains consistent at 0.17s for both implementations, suggesting that the dynamic response characteristics are well-preserved. The overshoot in HDL SIC (15.1%) is slightly higher than MATLAB SIC (14.68%), with a 2.86% deviation, which may require further fine-tuning in the hardware implementation. The steady-state value remains closely matched, with only a 0.52% difference, confirming that both implementations achieve similar final outputs. The steady-state time is slightly shorter in HDL SIC than in MATLAB SIC, reflecting a 3.36% improvement in settling time. Notably, the steady-state error is significantly lower in HDL SIC (0.4) compared to MATLAB SIC (0.6), demonstrating a 33.3% reduction, which suggests better accuracy in the HDL-based implementation. This discrepancy occurs because the generated data is converted into digital values and does not account for the pointer values. Overall, the findings indicate that HDL co-simulation provides a viable hardware-based alternative to MATLAB SIC, maintaining comparable performance while offering advantages in response time and error reduction. However, slight variations in overshoot highlight areas for potential optimization in hardware implementation.

Table 5. The value of System Identification Controller output response in MATLAB Simulink

	System Identification Controller of Knee Response		
	MATLAB SIC	***HDL SIC***	***Percentage Error (%)***
Time Delay	0.26s	0.24s	7.69
Rise Time	0.17s	0.17s	0
Overshoot	14.68	15.1	2.86
Steady-State Value	44.37	44.6	0.52
Steady-State Time	2.38s	2.3s	3.36
Steady-State Error	0.6	0.4	33.3

CONCLUSION

In conclusion, this study successfully achieved its primary objective of designing a digital system identification controller (SIC) for adaptive feedback control (AFC) in closed-loop Functional Electrical Stimulation (FES) systems. The SIC was implemented using Verilog HDL and tested through RTL simulations in Quartus and ModelSim. The results demonstrate that the adaptive feedback controller facilitates real-time self-tuning and adjustment of internal control settings, addressing patient-specific conditions such as muscle fatigue, time delay response, stiffness, and spasticity. The integration of the HDL SIC with MATLAB Simulink for co-simulation showed consistency in output accuracy, with percentage errors ranging from 0% to 33.3%, verifying the system's reliability. Moreover, the digital SIC's ability to extract key features—such as time delay, rise time, steady-state values, and overshoot—ensures precise adjustments to feedback control, enhancing the closed-loop FES system's adaptability and effectiveness. These findings underline the SIC's potential as a robust solution for overcoming nonlinear effects in muscle response, supporting extended rehabilitation activities for spinal cord injury patients.

ACKNOWLEDGMENT

The authors express their gratitude to the Ministry of Higher Education (MOHE) Malaysia and Universiti Teknologi MARA, Cawangan Pulau Pinang, for providing the research facilities used during the experimental work. The APC and research activities were funded by MOHE Malaysia through the Fundamental Research Grant Scheme (FRGS) under grant number FRGS/1/2021/TK0/UITM/03/1.

REFERENCES

[1] Lynch, C.L. and M.R. Popovic, *A Comparison of Closed-Loop Control Algorithms for Regulating Electrically Stimulated Knee Movements in Individuals With Spinal Cord Injury*. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 2012. 20(4): p. 539-548.

[2] Arcolezi, H.H., et al., *RISE controller tuning and system identification through machine learning for human lower limb rehabilitation via neuromuscular electrical stimulation*. Engineering Applications of Artificial Intelligence, 2021. 102.

[3] Noorsal, E., et al., *Design of an fpga-based fuzzy feedback controller for closed-loop fes in knee joint model*. Micromachines, 2021. 12(8).

[4] Saharul Arof, E.N., Saiful Zaimy Yahaya, Zakaria Hussain, Yusnita Mohd Ali 1, Mohd Hanapiah Abdullah, Muhamad Khuzzairie Saf, *Adaptive Sliding Mode Feedback Control Algorithm for a Nonlinear Knee Extension Model*. Machines, 2023. 11(7).

[5] Li, M., et al. *Adaptive Sliding Mode Control of Functional Electrical Stimulation (FES) for Tracking Knee Joint Movement*. in *Proceedings - 2017 10th International Symposium on Computational Intelligence and Design, ISCID 2017*. 2017.

[6] Li, Z., et al., *Real-Time Closed-Loop Functional Electrical Stimulation Control of Muscle Activation with Evoked Electromyography Feedback for Spinal Cord Injured Patients*. International Journal of Neural Systems, 2018. 28(6).

[7] Razavian, R.S., et al., *Feedback Control of Functional Electrical Stimulation for 2-D Arm Reaching Movements*. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 2018. 26(10): p. 2033-2043.

[8] Izci, D., et al., *Multi-strategy modified INFO algorithm: Performance analysis and application to functional electrical stimulation system*. Journal of Computational Science, 2022. 64.

[9] Jezernik, S., R.G.V. Wassink, and T. Keller, *Sliding Mode Closed-Loop Control of FES: Controlling the Shank Movement*. IEEE Transactions on Biomedical Engineering, 2004. 51(2): p. 263-272.

[10] Arof, S., et al., *Pole placement tuning of proportional integral derivative feedback controller for knee extension model*. Indonesian Journal of Electrical Engineering and Computer Science, 2024. 34(3): p. 1566-1581.

[11] Tan, W., W. Yu, and H. Wang, *Dynamic Event-Triggered Integral Sliding Mode Adaptive Optimal Tracking Control for Uncertain Nonlinear Systems*. Symmetry, 2022. 14(6).

[12] Wang, L., et al., *Implementation of integral fixed-time sliding mode controller for speed regulation of PMSM servo system*. Nonlinear Dynamics, 2020. 102(1): p. 185-196.

[13] Muhan, N.H.M., et al. *Design of Digital System Identification Controller for a Nonlinear Knee Model in Closed-Loop Functional Electrical Stimulator (FES)*. in *2024 IEEE 14th International Conference on Control System, Computing and Engineering (ICCSCE)*. 2024.

[14] Kong, N., et al. *Automated System Identification of Digitally-Controlled Multi-phase DC-DC Converters*. in *2009 Twenty-Fourth Annual IEEE Applied Power Electronics Conference and Exposition*. 2009.

[15] Sun, X., L. Zhang, and J. Gu, *Neural-network based adaptive sliding mode control for Takagi-Sugeno fuzzy systems*. Information Sciences, 2023. 628: p. 240-253.

[16] Previdi, F. and E. Carpanzano, *Design of a gain scheduling controller for knee-joint angle control by using functional electrical stimulation*. IEEE Transactions on Control Systems Technology, 2003. 11(3): p. 310-324.

[17] Wos, P. and R. Dindorf, *Modeling and Identification of the Hydraulic Servo Drive*. EPJ Web of Conferences, 2019. 213: p. 02100.

[18] Arun Prasad, K.M. and U. Nair, *Intelligent fuzzy sliding mode controller based on FPGA for the speed control of a BLDC motor*. International Journal of Power Electronics and Drive Systems, 2020. 11(1): p. 477-486.

[19] Kung, Y.S. and M.H. Tsai, *FPGA-Based Speed Control IC for PMSM Drive With Adaptive Fuzzy Control*. IEEE Transactions on Power Electronics, 2007. 22(6): p. 2476-2486.