IMPLEMENTATION AND VALIDATION OF A PID-CONTROLLED MASTER-SLAVE EXOSKELETON ARM FOR PRECISION ELBOW REHABILITATION

Muhammad Zulhilmi Hussin[1], Jamaludin Jalani[2], Wan Ahmad Khairuddin Wan Zakaria[3], Amirul Syafiq Saadun[4]

ABSTRACT

Weakness in the arm after a stroke often makes everyday tasks hard to manage. Because robotic sleeves can guide repeated motions, they may help regain movement over time. A new approach is presented in which one limb takes charge and the second follows, tied together by a digital twin that adjusts motion with rapid response control. Instead of operating independently, the stronger side sets the movement while the weaker side copies it, with real-time corrections. Trials were conducted twice—first in simulation, then again on physical hardware to repeat the test. In the open-loop condition, the tracking error was substantial, reaching a maximum deviation of 76%. Once PID control was introduced, the error was reduced to thirteen digits calm Overshoot was also reduced, to approximately four or five percent. The system settled more quickly, with improved response. Building models in MATLAB Simulink demonstrated the feasibility of the system, with each component controlling the elbow joint smoothly. Tuning the controller parameters ensured that motion remained fluid across repeated trials. This kind of linked robotic arm, with a leader guiding a follower, might support long-term rehabilitation at home or in remote clinics. In future work, researchers plan to integrate more advanced control strategies and evaluate their real-world impact on people recovering from stroke.

Keywords: exoskeleton arm, elbow rehabilitation, PID control, bilateral training, master-slave system, MATLAB Simulink, stroke therapy

1. INTRODUCTION

One major reason people struggle after a stroke is difficulty voluntarily moving their arms, which may be severe in some cases. Because the elbow does not function properly, reaching or lifting can become difficult, making everyday activities more challenging. Instead of relying only on standard rehabilitation

[1] Assistant Professor, Department of Mathematics, Jashore University of Science and Technology, Jashore, Bangladesh; PhD Student, Department of Mathematics, The Chinese University of Hong Kong, Hong Kong. Email: mnazrul.math@just.edu.bd
[2] Associate Professor, Department of Electronic, Faculty of Electrical and Electronic Engineering, UTHM, 86400, Batu Pahat, Johor, Malaysia +60167779004, Email: jamalj@uthm.edu.my
[3] Researcher, Department of Electronic, Faculty of Electrical and Electronic Engineering, UTHM, 86400, Batu Pahat, Johor, Malaysia +60167779004, Email: he240024@student.uthm.edu.my
[4] Associate Professor, Department of Electrical Engineering, Faculty of Engineering Technology, UTHM, Hab Pendidikan Tinggi Pagoh KM 1, Jalan Panchor 84600 Panchor, Johor, Malaysia +601116602509, Email: amirul@uthm.edu.my

Manuscript received on 15 April 2026, revised on 03 July 2026 and accepted on 07 July 2026 as per publication policies of *NED University Journal of Research*. All rights are reserved in favour of NED University of Engineering & Technology, Karachi, Pakistan.
This article is published as open access and is distributed under the terms of the Creative Commons Attribution 4.0 International License (CC BY 4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are properly credited.

https://doi.org/10.35453/NEDJR-Icon3E2025-006-R1

methods, engineers have developed robotic braces that help restore movement. These devices carefully guide the arm through repeated motions to support the recovery of motor control. When intelligent control strategies are incorporated into these machines, adjustments can be made smoothly based on the user's movement. A different approach to movement support involves linking two limbs so that one guides the other during recovery. When the healthy arm or limb leads the motion, therapy can become more balanced and active. Still, if timing slips or movement paths drift apart, coordination can deteriorate under shifting loads or external forces. Systems that send signals forward without checking the results may miss their targets, resulting in overshoot, delayed responses, or inaccurate positioning. Instead of operating without feedback, continuous error monitoring allows constant adjustments. Here, a method using proportional, integral, and derivative corrections helps fine-tune responses through continuous monitoring.

The PID controller was chosen because it handles disturbances well while reducing steady-state errors without excessive overshoot. Testing was conducted in two ways: first through simulations in MATLAB and then directly on physical equipment to evaluate the behavior of each setup. The first setup ran freely, the second was locked but unadjusted, and the third used a tuned PID loop. This study presents the development and validation of the robotic arm control system, focusing on the specific objectives of the research.

2. RELATED WORK

Machines that help people move again have become more common, mainly because they offer steady, measurable support during recovery from movement problems. Such tools can deliver precise and repetitive motion without fatigue. One type of device, worn on the arm, has helped patients recovering from stroke regain movement and strength through repeated practice. Instead of guessing progress, therapists see real data from each session. While results differ, many find better control when using these devices regularly. These devices commonly focus on elbow flexion, forearm movement, and the gradual restoration of lost function.

Getting the elbow moving well again matters a lot when regaining arm function. Devices like MIT-Manus and ARMin help by guiding patients through bending and straightening motions, offering powered support along the way [1], [2]. Instead of fixed settings, they adapt - giving more or less push depending on how far someone has come. Cost becomes a barrier though; many people cannot afford them, especially for use at home. That keeps these tools out of reach despite how useful they might be. Most of these gadgets struggle to track movement precisely - particularly for people whose muscles respond unpredictably. Because of that, effective control systems now must be built right into exoskeletons, making them safer and more effective at the job they do.

One approach to rehabilitation uses a setup in which one arm guides the other during therapy. When the unaffected arm moves, it guides the affected arm in a synchronized manner. Research suggests that bilateral movement training can support motor recovery following stroke [3]. Repeated symmetrical movements may promote coordination between the limbs and improve motor control [4]. However, achieving precise coordination between two components remains a common challenge in master–follower configurations, where delays and trajectory misalignment can occur [5], [6]. Real-time feedback can improve synchronization between the limbs and reduce motion drift [7], [8].

PID control is widely used in robotic systems because it can continuously adjust the system response based on the control error [9]. Although relatively simple, the PID approach remains effective, particularly for systems with moderate control complexity. Tuning the individual PID components allows engineers to adjust the system response based on the current error and its accumulated and changing behavior. In rehabilitation wearables, this combination can help maintain the desired motion while reducing oscillations and positioning errors. The resulting response can remain close to the desired trajectory under varying operating conditions [10].

Mavroidis and colleagues [11] used an exoskeleton with PID control to help people move their elbows properly again, making it easier to follow correct joint positions. Moreover, Hussain and colleagues [12] explain that combining the exoskeleton with PID control can reduce muscle effort during therapy movements. In addition, new techniques such as the Ziegler-Nichols rules use computer-based methods that combine fuzzy logic with standard PID designs, with the aim of achieving faster responses without interruptions [13], [14], [15]. By focusing on the three main values of K_p, K_i and K_d, determining the exact values can be challenging because body motion can vary spontaneously and disturbances may occur unexpectedly during real-time operation.

Some research has explored PID tuning through simulations or trial-and-error techniques, leading to clear gains in how well systems operate [15]. When open-loop and closed-loop setups are measured against each other, results often favor PID controls - especially in rehab equipment. Systems guided by PID usually show less long-term deviation, smaller peaks during response, along with quicker stabilization after disturbances. Under shifting loads, their behavior stays steadier compared to alternatives. Because of these traits, engineers lean toward embedding PID logic into robotic therapy tools where precision and reliability matter most. Price tags and access hurdles still block broad use of rehab robots. These days, researchers have started using 3D-printed components [16], along with freely shared blueprints such as those based on Arduino or ESP32 development modules and low-cost sensors, helping cut expenses without losing performance. Instead of custom hardware, studies by Wang's lab and separately by Tagliamonte's crew found that regular servomotors combined with plastic joints made from PLA have demonstrated suitability in exoskeletons [17]. Real-time tracking plays a key role in rehabilitation sessions [18]; it helps prevent injury while ensuring patients stick to prescribed movements [19], [20]. When testing affordable robotic systems quickly, platforms such as LabVIEW appear frequently, just like MATLAB-Simulink, particularly when actual devices run within simulated environments.

Although robotic rehabilitation systems and PID-based control strategies have been investigated separately, their integration into a single affordable system capable of real-time operation remains limited. In particular, the integration of master–slave mirrored motion, PID-based control, and comparative evaluation under simulated and experimental conditions requires further investigation. Therefore, this study develops and evaluates an elbow rehabilitation exoskeleton in which a controlled arm follows the motion of a lead arm using PID control. The system performance is evaluated by comparing configurations with and without feedback control under both simulation and experimental conditions.

3. METHODOLOGY

The powered arm brace was developed with mechanical components and control systems to monitor and support arm movement. The system consists of sensors and motors that enable one arm to follow the movement of the other during operation. When the healthy arm bends or straightens, the electronic system records the joint angle in real time. This data is transferred to the processing unit, which adjusts the supporting arm accordingly. Movement is controlled through successive corrections based on the tracking error. Instead of relying on predefined movements, the system uses continuous feedback to adjust the motion at each instant. A mathematical control rule determines the response of the system, helping to maintain the desired motion and reduce the difference between the expected and actual positions. Elbow movements are therefore maintained through continuous sensing and automatic adjustment.

3.1 System Overview

The developed robotic exoskeleton arm is designed to support elbow flexion and extension, with a motion range from 0° to approximately 180°. The system uses Feetech motors, while the frame is constructed from lightweight 3D-printed components. A sensor measures the joint angle and provides real-time feedback to maintain accurate and responsive movement. The system operates in a master-slave configuration, in which the healthy arm provides the reference movement and the affected arm follows the corresponding motion. The receiving arm adjusts its position through a feedback control loop. A microcontroller processes the signals between the sensors and actuators, while software tools support the overall system operation. MATLAB Simulink was used for system modelling, tuning, and simulation, while real-time implementation was achieved using hardware-in-the-loop control with Arduino and LabVIEW for data visualization.

3.2 System Modelling and Parameter Identification

Selecting appropriate system parameters is important for achieving suitable motion performance in an exoskeleton arm. Experimental input-output data were therefore used with the MATLAB System Identification Toolbox to determine the system parameters. The identified parameters were subsequently used to develop and evaluate the control system.

Step 1 Data Collection: First, input-output data was collected from the DC servo motor by applying a known input signal and recording the output position or speed of the motor over time as shown in **Figure 1.**

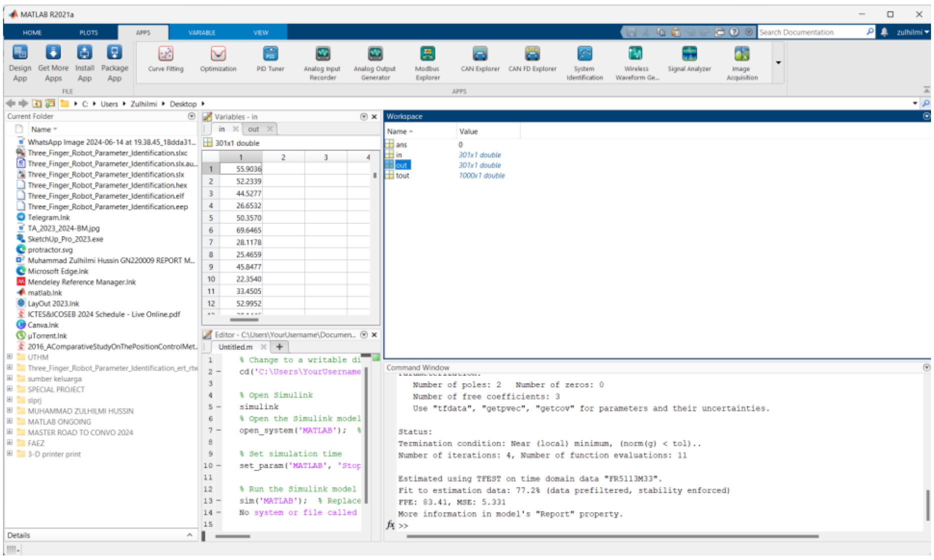

Figure 1. MATLAB R2021b interface with simulink model script and workspace data

Step 2: Load Data into MATLAB: The collected data was loaded into MATLAB in a suitable format, typically as time-series data.

Step 3: Importing Data into the System Identification Toolbox: The system identification toolbox was opened, and the data was imported through the Import Data option by selecting Time Domain Data. Then, load the iddata object created earlier, as shown in **Figure 2**.

Step 4: Model Estimation: The Estimate option was selected to estimate the system model. The desired model type, such as a transfer function, state-space, or ARX model, was then selected, and the corresponding model structure was specified. For the transfer function model, the number of poles and zeros was defined, and the appropriate continuous-time or discrete-time estimation option was selected. Finally, the Estimate button was used to obtain the system model based on the experimental data.

Step 5: Model Validation: The estimated model was validated by comparing its output with the actual measured output using the Compare option in the toolbox.

Step 6: Model Export: After satisfactory model performance was obtained, the identified model was exported to the MATLAB workspace for further analysis and control design.

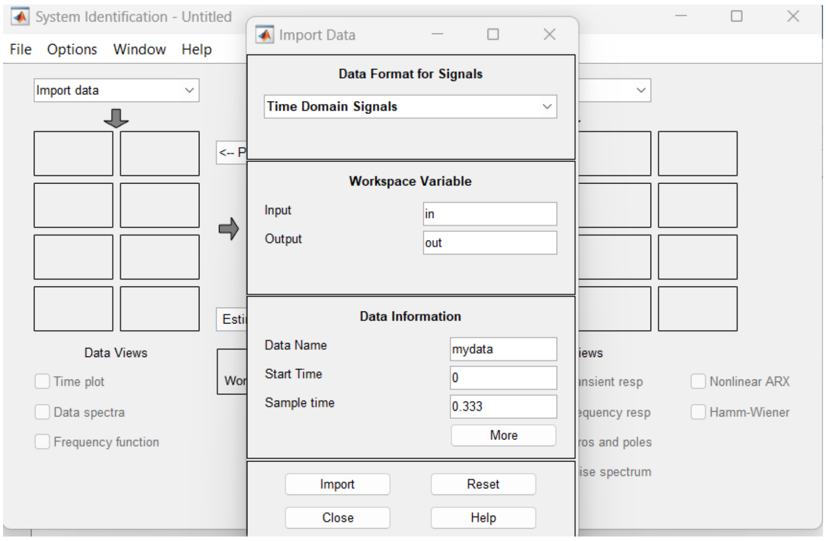

Figure 2. MATLAB system identification toolbox- import data dialog box

3.3 Control Strategy and PID Implementation

The adjustment was conducted to measure the difference between master and slave arm angles. The signal from controller is moved toward the motor to track the error and the reactions form from the controller. A PID controller was implemented to enable the motor to track the target angle while maintaining system stability. The controller content based on a few parameters need to focus based on the PID controller which is $u(t)$ equals k_p times $e(t)$, k_i multiplying with the integral of $e(t)$ over time, k_d scaling the rate of change in $e(t)$. The $u(t)$ functions are to detect the signal and moves straight into the actuator while $e(t)$ detects the degree of the joint angles. Moreover, the k_p also reacts with speed while k_i use to show the error and k_d role to soften the reaction. The changes are mostly based on observation during each test by repeating the small changed motion without any delays or overshoot. This is because the main focus is for the signal stability.

3.4 Block Diagram and System Flow

This study builds on a conventional PID setup and incorporates continuous user feedback as shown in **Figure 3**. When the robotic limb deviates from the intended position, a correction is initiated immediately after the deviation is detected. The difference, known as tracking error, triggers a corrective signal. The signal is sent directly to the motor to guide the motion back toward the target position. The PID controller consists of proportional, integral, and derivative terms, where the proportional term responds to the current error, the integral term accounts for past errors, and the derivative term responds to changes in the error. The resulting control signal, $u(t)$, derives the actuator, while $e(t)$ represents the difference between the actual and target joint positions. The proportional term provides a rapid response, the integral term reduces persistent offsets, and the derivative term helps counteract sudden changes. The controller parameters were tuned based on the observed motion behaviour to reduce overshoot and response delays. Stability was given priority throughout the tuning process. Even when the arm motion changed unexpectedly, the method maintained stable performance and remained straightforward to adapt during operation.

$$TF(s) = \frac{0.04232}{s^2 + 0.7116s + 0.07393} \tag{1}$$

Up to 0.004804 could affect the upper values, while the initial straight segment may shift by as much as 0.080695, followed by a variation of only 0.006935 near the base readings. **Eq. (1)** forms the basis of the controller implemented in MATLAB/Simulink. The real-time data flows through the block diagram, illustrating how the input signals are processed to generate the control actions required for the self-correcting PID setup. These control adjustments regulate the motion of the exoskeleton-style robotic limb with greater precision.

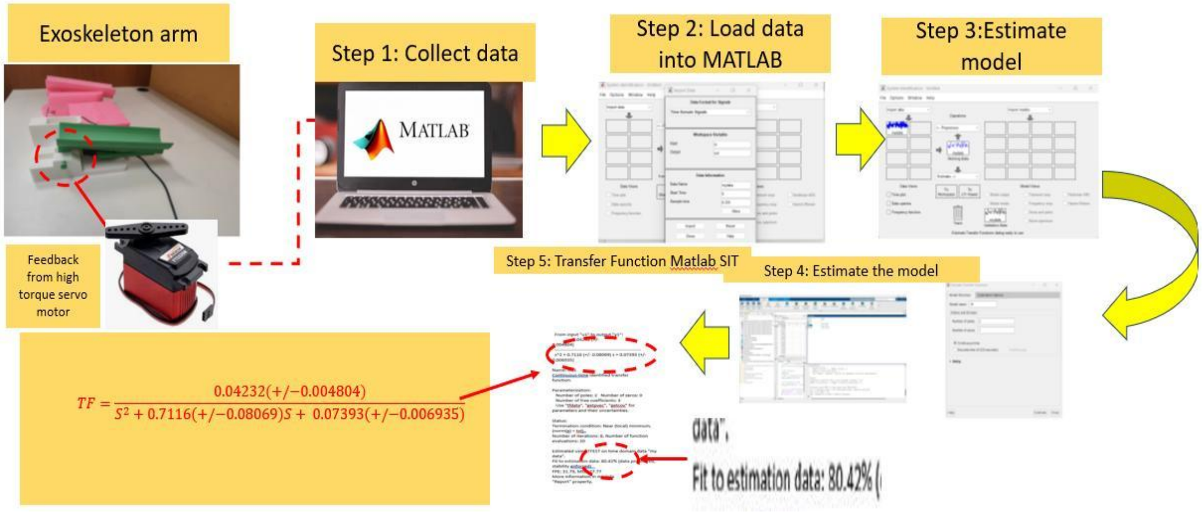

Figure 3. Block diagram of the PID-controlled master-slave exoskeleton arm

4. RESULTS AND DISCUSSION

The presence of the PID controller demonstrates the difference between the target position and the actual motor angle. The exoskeleton robotic arm setup consists of the developed components, including a DC servo motor, together with real-time monitoring of the actual movement. The resulting measurements provide feedback to the summing junction, thereby forming a closed loop. Each loop cycle works to improve accuracy by gradually reducing the error. At the beginning, the "In" block receives the reference signal, which is directed toward the "Out" block that represents the system output. The transmitted data are displayed directly on the Scope block, allowing the movement of the exoskeleton arm to be observed. The entire system operates continuously, with the output varying over time.

4.1 PID Controlled System Simulation

The block labelled "PID" receives the error, which is the difference between the actual and desired positions, and processes this error to generate a control signal that drives the motor toward the target position. The "Exoskeleton robotic arm plant" represents the system dynamics, incorporating components such as a servo motor and real-time feedback of the limb position. The feedback signal is returned to the summing junction, where it is compared with the reference input. This closed-loop process continuously works to reduce the error and improve positioning accuracy. The "In" section receives the input commands, while the "Out" section provides the system output for analysis or display. The subsystem output is then connected to a Scope component, enabling real-time monitoring of the system response. Thus, the movement of the exoskeleton arm can be observed as the input signals change over time.

The block parameters of the PID controller used in the Simulink model are shown in **Figure 4**. The controller is configured as a PID controller operating in continuous-time form with a parallel structure. The proportional gain is set to 8.1760, determining the response to the current error, while the integral gain is set to 26.07589, determining the response based on the accumulation of past errors. No derivative effect is present because the derivative gain is set to zero. When required, smoothing can be applied; however, the default filter coefficient is set to 100. The PID Tuner App was used to adjust the controller parameters. Zero-crossing detection is enabled to detect changes in the sign of the error. The system was tested at target positions of 40°, 90°, and 120°. After each step change, the system response was observed to determine how quickly the output reached and stabilized at the desired position.

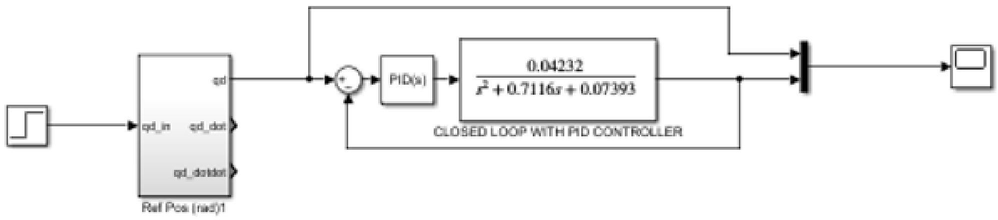

Figure 4. Simulink model of closed loop using PID controller simulation

Based on **Figure 5**, the red line appears to overlap the blue line. The blue line quickly reaches 40° and remains stable. In **Figure 6**, the target position is 90°. The system demonstrates a fast response with minimal overshoot and a smooth settling response. In **Figure 7**, the target position is 120°. The system also reaches the target position with a precise and stable response. Thus, each target angle demonstrates a similar and consistent response, indicating that the PID controller was able to stabilize the system at the specified target angles.

Based on **Table 1**, the performance of the reference and PID controllers is compared at three target angles: 40°, 90°, and 120°. The key parameters considered include overshoot, settling time, rise time, setpoint, and achieved angle. At 40°, the reference model exceeds the target by approximately 2%. The system takes approximately 8 seconds to fully stabilize. The rise time is approximately 2 seconds as the system transitions from the initial value to the target. Once the system settles, the final value remains stable. At 90°, the overshoot slightly increases to 2.1597%, with a settling time of 8.3327 seconds, a rise time of 2.00814 seconds, and zero steady-state error. For the 120° position, the overshoot is 2.1616%, the settling time is 8.3669 seconds, the rise time is 2.0797 seconds, and there is no steady-state error.

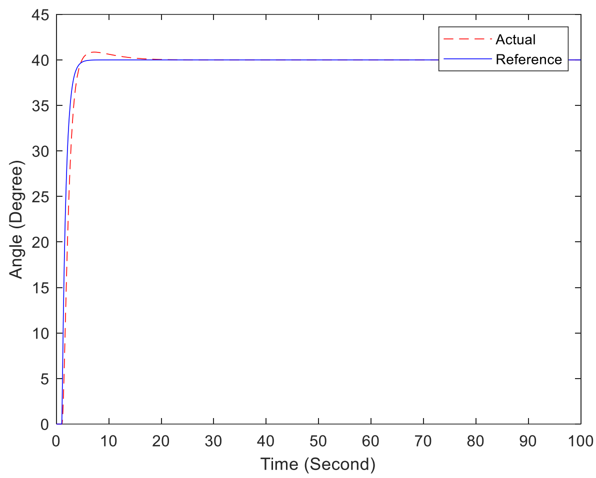

Figure 5. The closed loop PID controller step response at 40°

In contrast, the PID controller shows different performance characteristics. At 40°, the overshoot is 5%, the settling time is 19.12 seconds, the rise time is 2.007 seconds, and the steady-state error is 1.3500 x 10^{-13}. For the 90° position, the overshoot decreases to 4.4%, with a settling time of 20.45 seconds, a rise time of 3.003 seconds, and a steady-state error of 7.389×10^{-13}. At 120°, the PID controller has an overshoot of 4.2%, a settling time of 21.77 seconds, a rise time of 3.02 seconds, and a steady-state error of 1.0374×10^{-12}. The results demonstrate that the two control approaches exhibit different dynamic responses. The reference system maintains a relatively stable response, whereas the PID-controlled system exhibits greater overshoot before reaching the final value. The response characteristics also vary according to the target position, demonstrating the influence of the control method on the overall motion response.

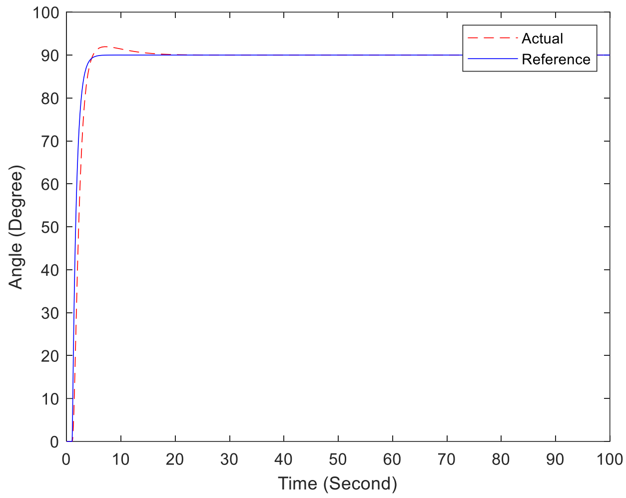

Figure 6. The step response of a PID controlled closed loop at 90°

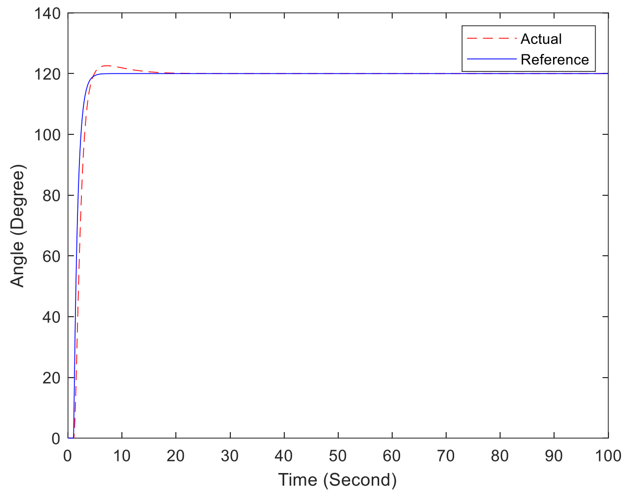

Figure 7. The closed loop PID controller step response at 120°

Table 1. PID-controlled simulation performance

Controllers	Setpoint (°)	Achieved (°)	Steady-State Error (°)	Overshoot (%)	Settling Time (s)
Reference	40	40	0	2.1437	8.3026
Reference	90	90	0	2.1597	8.3327
Reference	120	120	0	2.1616	8.3669
PID Controller	40	40	1.3500×10^{-13}	5.0000	19.1200
PID Controller	90	90	7.3896×10^{-13}	4.4000	20.4500
PID Controller	120	120	1.0374×10^{-12}	4.2000	21.7700

4.2 PID-Controlled Real-Time System

A Simulink setup is shown in **Figure 8**, representing a rehabilitation-focused robotic arm exoskeleton. The system is referred to as the “Exoskeleton robotic arm for rehabilitation” and uses a PID controller to improve its response. The Step1 block generates a step signal that represents the target position for the device. This input initiates the response of the entire control loop. The step input is connected to a transfer function block that represents the system dynamics and the desired behavior of the exoskeleton arm. A summing junction compares the desired position with the feedback signal representing the actual position of the exoskeleton arm. The resulting error is then fed into the PID controller to minimize the difference between the desired and actual positions.

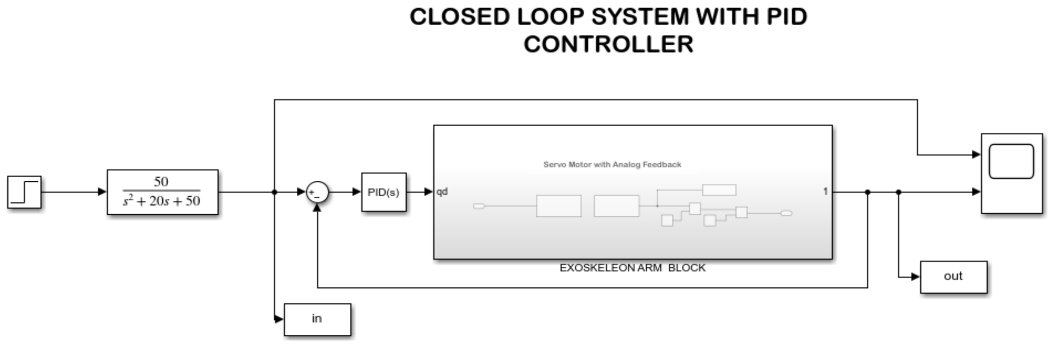

Figure 8. The closed-loop with PID controller step response for 120°

Figure 9 details the subsystem labelled "Servo Motor with Analog Feedback," which forms part of the exoskeleton robotic arm plant shown in the first image. The subsystem receives an input signal qd, representing the desired position. The signal is fed into an Arduino block (Pin 3) to control the servo motor. Feedback from the servo motor is obtained through another Arduino block (Pin A1), which reads the actual motor position. This feedback signal is then compared with the desired position by subtracting a constant value of 60 and dividing the result by a constant value of 2 to normalize the feedback value.

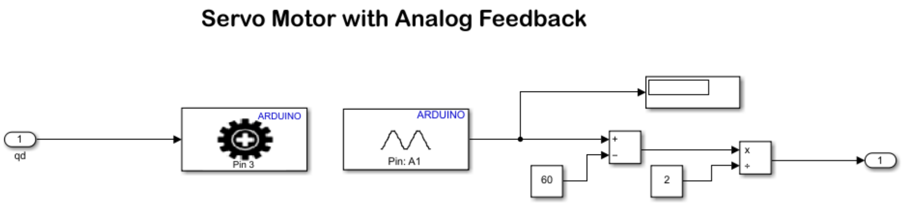

Figure 9. Exoskeleton arm simulation model

As shown in **Figure 10**, the controller operates in continuous time using a parallel PID configuration. The proportional gain is set to 8.1760 and responds to the current error. The integral gain is set to 26.0758 and accounts for accumulated past errors. Since the derivative gain is set to zero, no derivative action is applied. The filter coefficient is set to 100; however, it does not affect the system because the derivative term is inactive. The PID Tuner App was used to determine the controller parameters based on the system transfer function. Zero-crossing detection remains enabled throughout the operation to improve the detection of changes in the error signal. Within the closed loop, the step input establishes the target position, which passes through the mathematical model representing the system dynamics. The difference between the desired and actual positions is processed by the PID controller to generate the control signal for the robotic arm motor. The feedback signal continuously updates the controller output, allowing the system to reduce the positioning error and approach the desired trajectory. A Scope window displays the system response in real time, providing an indication of the overall control performance.

The system response was evaluated at three target positions: 40°, 90°, and 120°. In **Figure 11**, the real-time output, represented by the solid blue line, initially fluctuates and exceeds the target before gradually approaching the target represented by the red dashed line. This response indicates the behavior of the PID-controlled system during real-time operation. At 90°, as shown in **Figure 12**, the measured output initially exhibits overshoot before gradually approaching the desired level. Similarly, **Figure 13** presents the closed-loop response for a 120° target. The output initially exceeds the target and exhibits fluctuations before gradually approaching the desired position. These results demonstrate the real-time response of the PID-controlled exoskeleton arm to different target positions.

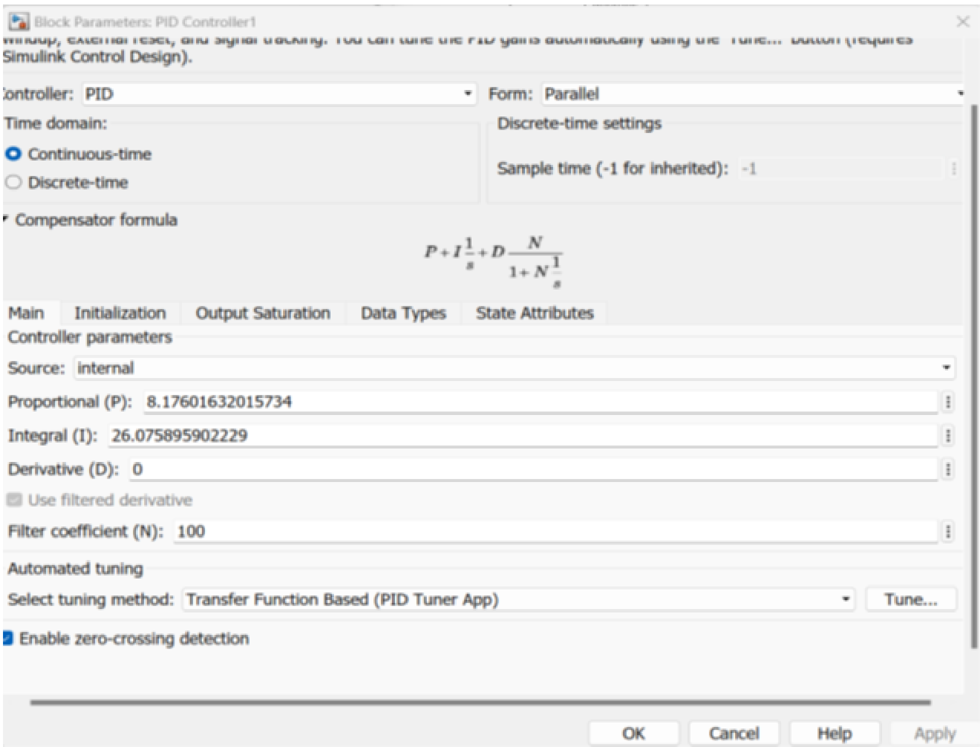

Figure 10. PID controller block parameters in simulink

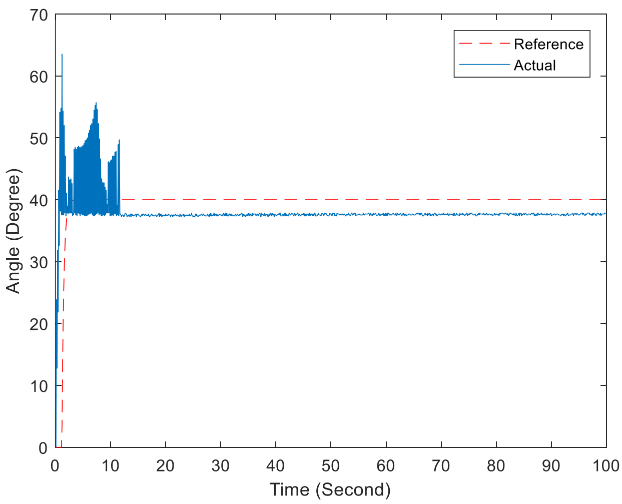

Figure 11. The PID-controlled system and uncontrolled system (real-time) step response for 40°

Table 2 compares the performance metrics of the reference system and the closed-loop PID-controlled system for different target positions (40°, 90°, and 120°). The key performance indicators include overshoot, settling time, rise time, and steady-state error. For the 40° position, the reference system shows no overshoot, a settling time of 2.4983 seconds, a rise time of 0.7462 seconds, and zero steady-state error. In contrast, the PID-controlled system exhibits an overshoot of 68.5714%, a settling time of 11.6937 seconds, a rise time of 0.4455 seconds, and a steady-state error of 2°. For the 90° position, the reference system maintains similar performance metrics, with no overshoot, a settling time of 2.4983 seconds, a rise time of 0.7462 seconds, and zero steady-state error. The PID-controlled system shows an overshoot of 47.3469%, a settling time of 10.6861 seconds, a rise time of 0.6190 seconds, and a steady-state error of 0.7407°. For the 120° position, the reference system continues to exhibit no overshoot, a settling time of 2.4983 seconds, a rise time of 0.7462 seconds, and zero steady-state error. The PID-controlled system demonstrates an overshoot of 76.4228%, a settling time of 11.0852 seconds, a rise time of 0.4439 seconds, and a steady-state error of 1.267°.

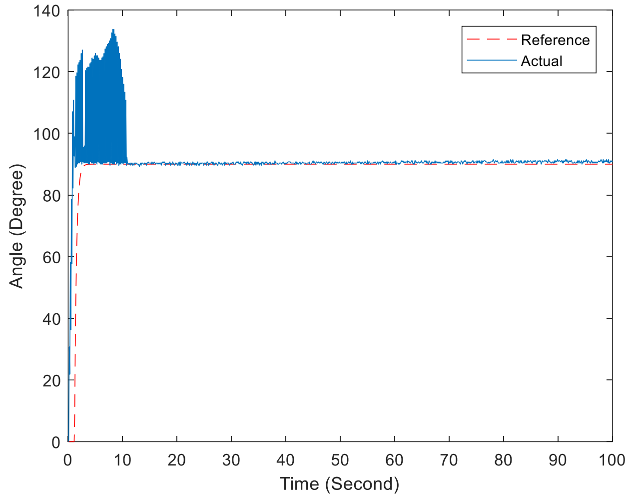

Figure 12. The PID-controlled system and uncontrolled system (real-time) step response for 90°

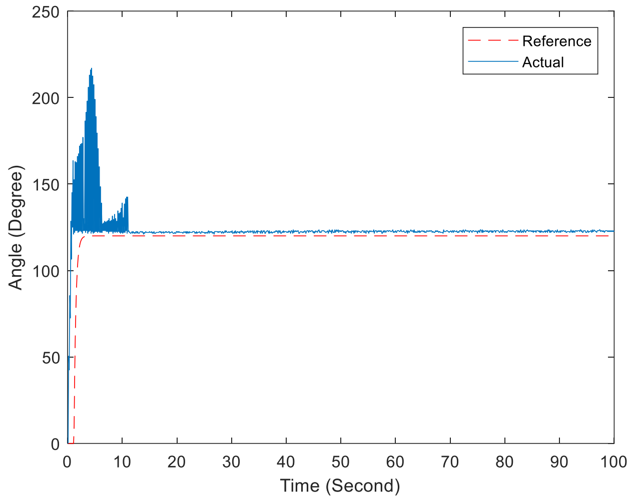

Figure 13. The PID-controlled system and uncontrolled system (real-time) step response for 120°

The results in **Table 2** demonstrate the significant influence of the PID controller on the system response. Although the PID-controlled system exhibits considerable overshoot during the initial response, it subsequently approaches the desired target. The overshoot may be attributed to the combined effects of the controller gains, motor inertia, and limited damping in the physical system. Further tuning of the proportional, integral, and derivative parameters may help reduce the overshoot and improve the overall transient response.

4.3 Comparison between Simulation and Real-Time PID Performance

The PID control method was evaluated through both simulation and real-time experiments at target positions of 40°, 90°, and 120°. Four performance parameters were considered: steady-state error, peak overshoot, settling time, and target-position accuracy. The simulation results are presented in **Table 1**, while the experimental results are presented in **Table 2**. In the simulation results, angle tracking remained very close to the target values, with residual errors ranging from 1.35×10^{-13} to 1.04×10^{-12}, while overshoot remained below 5%. The settling time ranged from 19.12 to 21.77 seconds. These results indicate accurate target tracking under simulated conditions. The real-time experiments produced less consistent results. Although the system generally approached the desired target angles, deviations were observed, particularly at higher target positions. The maximum overshoot reached 76.42% at the 120° target position. The settling time in the real-time experiments ranged from 10.69 to 11.69 seconds. Therefore, the real-time system exhibited faster settling but considerably greater overshoot than the simulation.

Table 2. Closed Loop system with PID controller (real-time)

Controllers	Setpoint (°)	Achieved (°)	Steady State Error (°)	Overshoot (%)	Settling Time (s)
Reference	40	38	0	0	2.4983
	90	90.02	0	0	2.4983
	120	120.01	0	0	2.4983
PID Controller	40	38	2	68.5714	11.6937
	90	90.02	0.7407	47.3469	10.6861
	120	120.01	1.267	76.4228	11.0852

The difference between the simulation and real-time results may be attributed to several practical factors, including sensor noise, servo motor dynamics, mechanical disturbances, hardware limitations, and signal-processing delays. These factors are not fully represented in the idealized simulation model and can affect the performance of the physical system. Therefore, further controller tuning and hardware-level improvements may be required to achieve more consistent real-time performance.

The use of a basic servo motor provides a cost-effective approach for initial development of the robotic arm. Such motors are widely available and relatively simple to integrate; however, their performance may limit the system as the complexity and requirements of the robotic arm increase. Future developments may therefore consider higher-performance actuators and more advanced control strategies. The use of a physical servo motor also provides a practical link between the simulation model and the real-world robotic system, which is particularly relevant for rehabilitation-assistive applications.

Although the system demonstrated functional performance during laboratory testing, further improvements may be required for reliable field operation. Future work could investigate adaptive control strategies or fuzzy-logic-based control to improve performance under varying operating conditions. Additional filtering techniques may also be applied to reduce the influence of sensor noise and sudden signal variations.

5. CONCLUSION AND RECOMMENDATION

This study aimed to develop an affordable robotic arm to assist in post-stroke rehabilitation. Instead of relying on costly equipment, the system used readily available components and was configured as two units, with each unit attached to an arm. During operation, motion data were exchanged between the two units, allowing one side to follow the movement of the other through a feedback control loop. The PID controller was implemented to improve motion control and positioning accuracy.

The simulation results demonstrated very small steady-state errors and relatively low overshoot for the tested target positions. However, the real-time experiments showed greater overshoot and variations in system response, highlighting the differences between ideal simulation conditions and practical hardware implementation. The results demonstrate the potential of a master-slave configuration for rehabilitation-oriented robotic applications, while also indicating the need for further controller tuning and hardware improvements.

Future work should focus on improving real-time control performance, particularly by reducing overshoot and improving response consistency. More advanced control strategies, improved sensors, and higher-performance actuators may be investigated to enhance the accuracy and reliability of the proposed system. These improvements could support the development of more practical and accessible robotic rehabilitation systems.

6. ACKNOWLEDGEMENT

This research was supported by the University of Tun Hussein Onn Malaysia through TIER1 Grant Scheme (Code Q487) and the Graduate Postgraduate Research Grant (GPPS Grant Code: J091) from Universiti Tun Hussein Onn Malaysia. The authors also thank the Faculty of Electrical and Electronic Engineering, UTHM, for providing the facilities and support throughout this study.

7. REFERENCES

[1] Ferraro M, Palazzolo JJ, Krol J, Krebs HI, Hogan N, Volpe BT. Robot-aided sensorimotor arm training improves outcome in patients with chronic stroke. Neurology. 2003; 61(11):1604-1607. doi:10.1212/01.WNL.0000095963.00970.68
[2] Cauraugh S, Summers J, Kim JK, Liao DB, Thigpen JJ. Bilateral arm training and stroke rehabilitation: a systematic review and meta-analysis. J Neuroeng Rehabil. 2005;2(1):13. doi:10.1186/1743-0003-2-13
[3] Zahari SNM, Shah NSM, Sadun AS, Lurdsuami S. Analysis of Flexitank heating pad data temperature using LabVIEW. Prog Eng Appl Technol. 2022; 3(2):354-362. doi:10.30880/peat.2022.03.02.035
[4] Whitall CE, McCombe Waller JS, Silver A, Macko R. Bilateral arm training: why and who benefits? NeuroRehabilitation. 2008; 23(1):29-41. doi:10.3233/NRE-2008-23104
[5] Michmizos KP, Rossi S, Castiello U, Krebs HI. Virtual reality and a haptic master-slave set-up in post-stroke upper-limb rehabilitation. J Neuroeng Rehabil. 2015; 12:59. doi:10.1186/s12984-015-0044-y
[6] Ramos-Murguialday A, Broetz D, Rea M, Läer L, Yilmaz Ö, Brasil FL, et al. Brain-machine interface in chronic stroke rehabilitation: a controlled study. Ann Neurol. 2013; 74(1):100-108. doi:10.1002/ana.23879
[7] Ranzani R, Zollo L, Krebs HI. Closed-loop task-difficulty adaptation during virtual-reality reach-to-grasp training post-stroke. Front Neurosci. 2016; 10:518. doi:10.3389/fnins.2016.00518
[8] Shahbazi M, Atashzar SF, Tavakoli M, Patel RV. Robotics-assisted mirror rehabilitation therapy: a therapist-in-the-loop assist-as-needed architecture. IEEE/ASME Trans Mechatronics. 2019; 24(4):1358-1369. doi:10.1109/TMECH.2019.2907812
[9] Saidi I, Touati N, Ben Hadj Mbarek H. PID control of a lower limb rehabilitation exoskeleton. Prz Elektrotechn. 2024; 100(6):60-67. doi:10.15199/48.2024.06.60
[10] Ahmed T, Islam MR, Brahmi B, Rahman MH. Robustness and tracking performance evaluation of PID motion control of 7 DoF anthropomorphic exoskeleton robot assisted upper limb rehabilitation. Sensors. 2022; 22(10):3747. doi:10.3390/s22103747
[11] Ziegler JG, Nichols NB. Optimum settings for automatic controllers. Trans ASME. 1942; 64(11):759-768. doi:10.1115/1.4019264
[12] Gull MA, Bai S, Bak T. A review on design of upper limb exoskeletons. Robotics. 2020; 9(1):16. doi:10.3390/robotics9010016
[13] Deb S, Paul M, Chakraborty A. Arduino-based prototype for low-cost elbow exoskeleton rehabilitation. Sens Actuators A Phys. 2016; 245:48-55. doi:10.1016/j.sna.2016.01.013
[14] Zhou Y, Wang M, Cheng T. PID versus model predictive control in upper limb assistive devices. Mechatronics. 2019; 56:123-131. doi:10.1016/j.mechatronics.2018.12.007
[15] Liu J, Fang H, Xu J. Online adaptive PID control for a multi-joint lower extremity exoskeleton system using improved particle swarm optimization. Machines. 2022; 10(1):21. doi:10.3390/machines10010021

[16] Ivanov MD, Fedulov AA, Volkov VA. Development of a low-cost 3D printed upper limb exoskeleton for stroke rehabilitation. J Rehabil Assist Technol Eng. 2022; 9:20556683221104523. doi:10.1177/20556683221104523
[17] Ruiz AT, Paredes-Madrid L, Martinez EP. Design of a 3D printed modular exoskeleton for upper limb rehabilitation in post-stroke patients. Sensors. 2021; 21(24):8286. doi:10.3390/s21248286
[18] Bodo G, Tessari F, Buccelli S, Laffranchi M. A rapid control prototyping and hardware-in-the-loop approach for upper limb robotic exoskeletons control. Appl Sci. 2024; 14(5):2017. doi:10.3390/app14052017
[19] Mejia-Trujillo J, Jimenez P, Rojas-Dominguez A, Rangel-Magdaleno J, Ramirez-Cortes JM. Design and implementation of a low-cost upper limb exoskeleton with EEG-based intention detection for stroke rehabilitation. Sensors. 2023; 23(23):9846. doi:10.3390/s23239846
[20] Wu J, Yao Y, Shang X. Causes and preventive measures for fire-related injuries in intensive care units: a systematic review. J Glob Health. 2025; 15:04043. doi:10.7189/jogh.15.04043

AUTHORS' PROFILES

Muhammad Zulhilmi Hussin *is currently pursuing a Master's degree in Engineering Technology at Universiti Tun Hussein Onn Malaysia (UTHM), with a background in Electrical Engineering. My research interests focus on IoT-based monitoring systems, real-time data acquisition, and the integration of human factors into industrial performance evaluation.*

Dr. Jamaludin bin Jalani *is an Associate Professor at the Department of Electronic, Faculty of Electrical and Electronic Engineering (FKEE), Universiti Tun Hussein Onn Malaysia. He holds a PhD in Humanoid Robot Engineering from the University of Bristol, an MSc in Mechatronics Engineering from International Islamic University Malaysia, and a Bachelor's degree in Electrical and Electronics Systems Engineering from Leeds Metropolitan University. His research interests include robotics, mechatronics, automation systems, intelligent control, human-robot interaction, rehabilitation exoskeletons, and engineering technology education. He also has industrial experience as an Engineer in Sharp Manufacturing Malaysia.*

Wan Ahmad Khairuddin bin Wan Zakaria *is currently pursuing a Master of Electrical Engineering at Universiti Tun Hussein Onn Malaysia under the Faculty of Electrical and Electronic Engineering. He previously obtained a Bachelor of Science (Hons.) in Computational Mathematics from Universiti Teknologi MARA with First Class honours and a CGPA of 3.70. His academic background combines strong analytical, mathematical, and engineering foundations. His areas of interest include electrical systems, automation, problem solving, data analysis, and applied computational techniques. With multidisciplinary knowledge in mathematics and engineering, he is well positioned to contribute to research, innovation, and technology-driven industries.*

Dr. Amirul Syafiq bin Sadun *is an Associate Professor (Ir., Ts.) at the Department of Electrical Engineering Technology, Faculty of Engineering Technology, Universiti Tun Hussein Onn Malaysia. He holds a PhD in Engineering Technology, a Master's in Technical Education (Electrical Engineering), and a Bachelor's degree in Electrical Engineering from Universiti Tun Hussein Onn Malaysia. His research interests include robotics, mechatronics, robot control, humanoid robots, signal processing, autonomous systems, and industrial automation. He is actively involved in research grants, publications, innovation projects, and academic leadership, serving as a Focus Group Leader and Principal Researcher in the faculty.*