System Classification of Hand Movements in Hand Prosthesis Prototype Based on Model Machine Learning Control

I Wayan Widhiada*, I Made Widiyarta, I Made Esa Dharmayasa Putra, I Gede Putu Agus Suryawan, Dewa Ngakan Ketut Putra Negara, and Tjokorde Gede Tirta Nindhia

Department of Mechanical Engineering, Faculty of Technology, University of Udayana, Badung, Indonesia
Email: wynwidhiada@unud.ac.id (I.W.W.); m.widiyarta@unud.ac.id (I.M.W.);
esa.darmayasa@gmail.com (I.M.E.D.P.); agus88@unud.ac.id (I.G.P.A.S.); devputranegara@unud.ac.id (D.N.K.P.N.);
tirta.nindhia@me.unud.ac.id (T.G.T.N.)
*Corresponding author

***Abstract*—Numerous investigations have explored automated control mechanisms for artificial hands; nonetheless, the performance of individual robotic fingers remains suboptimal. To enhance motion precision, this study proposes a model of hand movement control utilizing Artificial Intelligence (AI). Selecting suitable sensing components and applying specialized computational algorithms are key factors in optimizing the prosthetic control framework. A control architecture based on machine learning was developed, enabling the hand prosthesis prototype to perform movements automatically according to the classifications generated by the trained model. Experimental results demonstrate that increasing the neuron count in the learning model enhances predictive accuracy while minimizing loss values. A comparison between Artificial Neural Network (ANN) and Recurrent Neural Network–Long Short-Term Memory (RNN-LSTM) architectures revealed that the ANN configuration with a 32-neuron hidden layer provided the best results, achieving a loss below 0.1 and accuracy above 90%. Therefore, this ANN model was chosen as the main control algorithm for the prosthetic system. When implemented in the master controller, the model produced prediction accuracies exceeding 90% across all output classes, successfully activating the prosthetic hand according to ten predefined motion labels.**

***Keywords*—intelligent control, prosthetic hand, motion coordination, artificial neural network, sensor feedback**

I. Introduction

Individuals who experience arm or hand amputation must adapt to new physical limitations in performing daily activities. These limitations often influence their psychological state, particularly among those who have suffered severe trauma and are still struggling to accept their condition as persons with disabilities [1]. Previous studies indicate that the prevalence of depression among post-amputation patients ranges from 20.6% to 63%, while anxiety levels are reported between 25.45% and 57% [2]. Hence, psychological interventions play a crucial role in supporting mental recovery. A socially supportive environment is also essential to help patients regain confidence and emotional stability after limb loss.

Functional rehabilitation of the upper limb can further aid patients in achieving a sense of physical completeness. To enhance the quality of life for individuals with upper limb amputation, continuous advancements have been made in prosthetic technology aimed at restoring arm function. Early designs of mechanical prostheses were primarily intended to replace basic hand functions, such as gripping or holding objects using a hook mechanism. However, these devices required extensive manual control [3] and were often uncomfortable, leading to a relatively high rate of user rejection [4]. To address this issue, ergonomic improvements have been explored. A study by Daly *et al.* [5] highlighted that socket pressure distribution and precise fitting between the prosthesis and residual limb are key determinants of user comfort. Beyond ergonomics, the functionality of prosthetic arms has also become a major research focus. Many recent investigations have concentrated on the integration of Electromyography (EMG) [6–8] to enable automatic control. Nevertheless, both sensing methods still exhibit signal instability and variability among users, limiting their accuracy as input systems. Consequently, the selection of suitable sensors and the development of robust control algorithms are essential in designing prosthetic arms capable of producing movements that closely resemble natural limb motion.

In recent years, the use of Artificial Intelligence (AI) for myoelectric prosthesis control has advanced significantly. Several studies from 2022–2024 have proposed deep learning approaches to improve gesture recognition and proportional control. Chen *et al.* [9] presented a comprehensive survey of myoelectric control strategies, highlighting the transition from traditional feature-based methods toward deep neural architectures.

Manuscript received August 1, 2025; revised September 8, 2025; accepted October 31, 2025; published March 13, 2026.

doi: 10.18178/ijmerr.15.2.123-134

López *et al.* [10] introduced a hybrid CNN-LSTM model with post-processing to enhance gesture prediction from surface EMG, while Wang *et al.* [11] proposed MS-CLSTM, a multi-scale feature fusion and LSTM framework for robust manipulator control. Recent work by Williams *et al.* [12] further demonstrated the potential of recurrent convolutional networks for proportional and simultaneous control tasks. Joshi *et al.* [13] explored AI-enhanced EMG analysis for force and precision control in prosthetic applications. These developments underline the relevance of integrating deep learning into prosthesis control systems, and our work builds upon this trend by focusing on Referring to these problems, increasing the functionality of the control system aspect needs to be implemented. Thus, the manufacturing procedure can be formed to produce arm prosthesis products. The present study is expected to contribute to addressing these challenges by enhancing machine-learning-based control performance. Once these improvements are realized, post-amputation patients are expected to gain better access to functional prosthetic assistance, enabling a smoother return to normal daily activities [14].

II. Literature Review

Initially, upper limb (arm and hand) prosthetic devices for post-amputation patients primarily used the hook principle, where the prosthesis functioned mainly as a hook for picking up objects. However, only 62%–64% of patients have hand sizes that match this type of prosthesis. The inadequate ergonomic design of the hook prosthesis has led to a high rejection rate of approximately 60%. To date, most arm prostheses still rely on manual control systems. As a result, there has been significant development and research into the automatic control of arm prostheses. Recent advancements focus on the use of EMG sensors, which detect electrical signals generated by muscle activity, paired with PID Control in MATLAB/Simulink [15]. These signals are processed through signal processing techniques to produce the desired arm movement. EMG sensors have the advantage of being surface-mounted on the skin, near the site of amputation [16]. However, their practical use remains inconsistent, and conventional control systems have struggled to reliably achieve the desired output.

To address these limitations, researchers have proposed alternative Human-Machine Interfaces (HMIs) based on Force Sensitive Resistors (FSRs)**,** which capture myoelectric force data more robustly than surface EMG signals. FSR-based HMIs typically consist of internal and external components designed to enhance signal stability, predictive accuracy, resistance to sweat and hair interference, wearability, and cost-effectiveness. Compared to surface Electromyography (sEMG), FSR-based systems provide more stable signal acquisition and improved prediction accuracy, thereby offering a more comprehensive solution for prosthetic control. These characteristics position FSR-based HMIs as a significant advancement in active prosthesis technology, enabling smoother, more intuitive, and user-centered human-machine interaction [17]. FSR leads the way in creating smooth, intuitive, and efficient prosthetic interfaces.

In further developments, the tactile sensor proposed by Huang and Wu [18] successfully provided feedback on the surface roughness of objects using a Convolutional Neural Network (CNN) model. This mechanism has the potential to determine the arm's response to the surface texture of an object with a similar value. The application of CNN allows for more precise movement results. Furthermore, this output can be further optimized by combining the CNN with Artificial Neural Network (ANN) model, which offers more suitable computing capabilities than CNN alone.

In parallel, advances in manufacturing processes have played a crucial role in improving the accessibility of arm prostheses. Additive Manufacturing (AM) techniques, such as stereolithography and Fused Deposition Modeling (FDM), have gained increasing attention due to their ability to fabricate complex three-dimensional structures efficiently. Material selection remains a critical factor, as prosthetic components must be lightweight, elastic, and capable of mimicking the surface characteristics of the human body. Thermoplastic Polyurethane (TPU) has been widely recognized for its ability to achieve realistic biomechanical properties. Marco *et al.* [19] emphasized that continuous innovation in prosthetic functionality and control systems is essential for enhancing post-amputation quality of life.

III. Methodology Research

The development of the manufacturing process is also important in increasing the availability of arm prostheses. In accelerating the manufacturing process, the AM method was introduced, especially the application of the stereolithography model in the FDM, with its ease in printing various complexities of three-dimensional products.

A. Design of Hand Prosthesis Control System

In this study, a hand prosthesis is designed to enable integrated finger movements through the application of machine learning. The prosthesis is equipped with drive components and an electrical system, supported by a machine learning–based control mechanism that functions as the decision-making unit [20]. The design is adapted from the InMoov open-source 3D-printed hand and forearm originally developed by Belford *et al.* [21]. This open-source design provides a platform for implementing ten output channels derived from the deep learning neural network system.

The Degree of Freedom (DoF) in a hand prosthesis refers to the number of independent movements that can be generated relative to the coordinate system governing its motion. The DoF of the prosthesis can be determined by counting the number of movable joints. In this design, the hand prosthesis possesses 15 DoF, with each of the five fingers comprising three DoF corresponding to flexion and extension at the joints [22].

This hand prosthesis is composed of three primary hardware components: the Master Controller, the Slave Sensor, and the Slave Servo. The software architecture is divided into two modules: the Sensor Reading System and the Deep Learning Neural Network (DLNN) System. Within the hardware configuration, the Master Controller employs a Raspberry Pi as the central computing device. As a compact yet high-performance microcomputer, the Raspberry Pi provides sufficient computational capacity to execute the DLNN. In this system, the Master Controller functions as the primary processing unit that executes the DLNN algorithm based on input from Force-Sensitive Resistor (FSR) sensors, thereby enabling intelligent decision-making for prosthetic control [23].

The Slave Sensor module consists of an Arduino Mega 2560 microcontroller connected to ten FSR402 sensors. The Arduino Mega 2560 is selected due to its higher number of input/output pins compared to other Arduino variants, allowing integration with multiple sensors. The FSR402 sensors are strategically positioned around the forearm to detect pressure signals generated by muscle contractions. The Slave Sensor is responsible for executing the Sensor Reading System, which acquires data from the FSR402 sensors and transmits the processed signals to the Master Controller. The Slave Servo module employs an ESP32 microcontroller connected to five MG996R servo motors. This unit receives motion commands from the Master Controller, based on the output of the DLNN System. The servos are mechanically linked to each finger of the prosthesis using tendon-like strings, enabling finger actuation and coordinated movement. For the DLNN System, two architectures are designed and implemented: the ANN and the Recurrent Neural Network-Long Short-Term Memory (RNN-LSTM). These models serve as the decision-making algorithms that map FSR input data to corresponding prosthetic finger movements [14].

In general, this study adopts an experimental research method to design, implement, and evaluate the proposed hand prosthesis system. The overall workflow of the system architecture is illustrated in the block diagram of the hand prosthesis control system, as presented in Fig. 1.

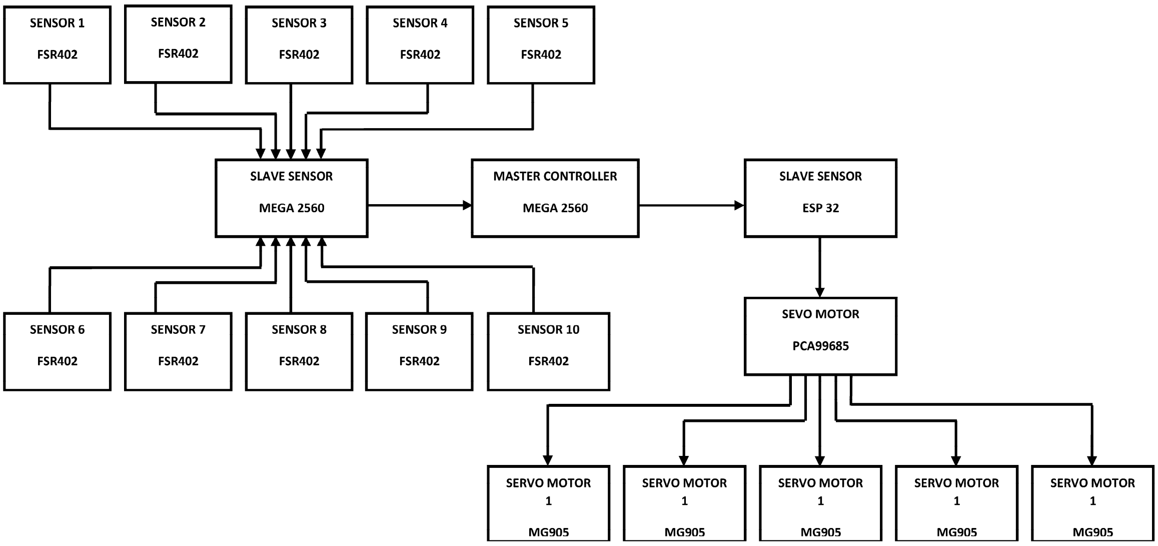


Fig. 1. Block diagram of hand prosthesis control system.

B. Design of a Force Sensor Reading System

The force sensor reading system is implemented through the Slave Sensor module. In this hand prosthesis design, ten FSR402 sensors are deployed and divided into two groups: the volar side and the dorsal side of the forearm, with five sensors positioned on each side. All force sensors operate simultaneously, and their data are transmitted collectively to the Master Controller (Raspberry Pi) via serial communication. The strategic placement of the FSR402 sensors enables the system to capture pressure variations resulting from muscle contractions in different regions of the forearm. The sensor configuration and positioning are illustrated in Fig. 2.

C. Deep Learning Neural Network System Design

In preparing a neural network model for deployment, it is essential to conduct a dataset collection phase followed by a machine learning–based training process. For this study, the dataset consists of hand movement data collected from non-amputated individuals, serving as the reference for normal hand function. Data acquired from the FSR402 sensors are labeled with corresponding movement signals to facilitate system design. The dataset comprises ten input channels, which are categorized into ten classes representing both individual and combined finger movements, as illustrated in Fig. 3. The performance of the machine learning models is evaluated using standard metrics, namely accuracy and loss values, recorded throughout the training process. These parameters provide insight into the model’s ability to learn movement patterns effectively and to generalize during subsequent deployment [14]. The performance of the machine learning models is evaluated using standard metrics, namely accuracy and loss values, recorded throughout the training process. These parameters provide insight into the model's ability to learn movement patterns effectively and to generalize during subsequent deployment.

The neural network architectures employed in this study are the ANN and the RNN-LSTM, both of which are well-suited for processing pattern recognition tasks such as those derived from force sensor data [24]. Training is conducted using datasets obtained from non-amputated (normal) hands, as these provide consistent and reliable ground truth for labeling finger movement patterns. This approach ensures that the models learn accurate mappings between sensor inputs and intended hand movements, which can then be transferred to prosthetic applications for post-amputation patients [14, 25].

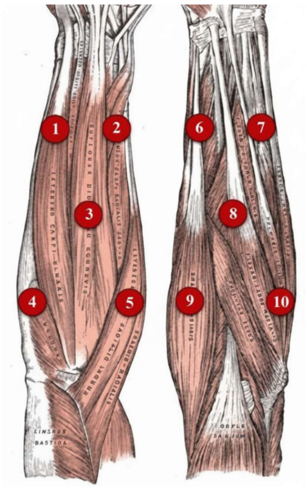


Fig. 2. Placement of the FSR402 sensor on the dorsal side and volar side.

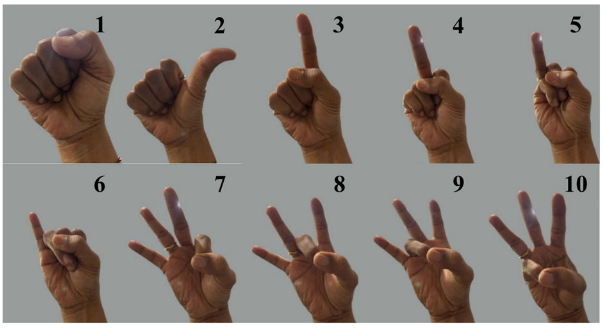


Fig. 3. Visualization of the ten classes of fingers.

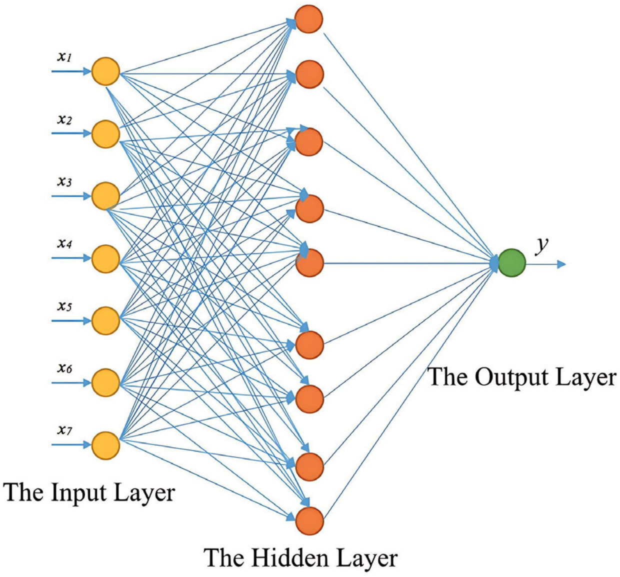


Fig. 4. Artificial neural network model architecture [26].

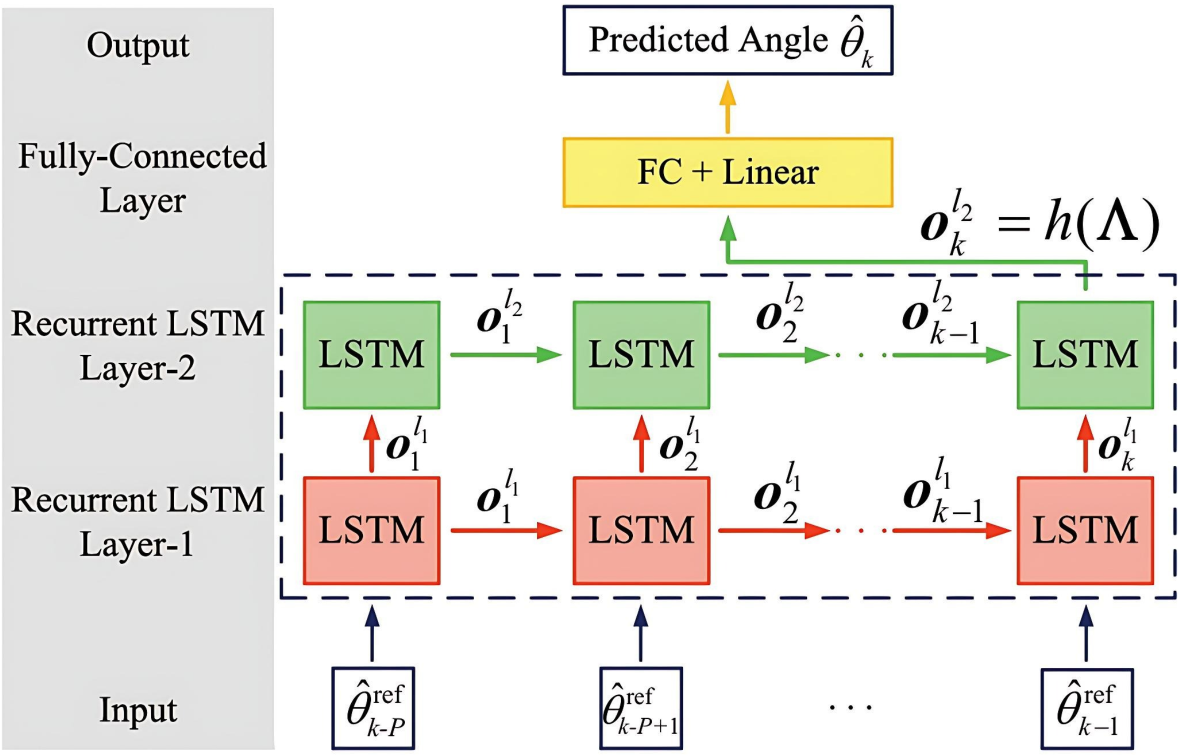


Fig. 5. Recurrent neural network model architecture-long short-term memory [28].

Several parameters are required to construct the deep learning neural network model for this system. The ANN architecture consists of three main components: the input layer, the hidden (fully connected) layer, and the output layer. The input layer contains ten neurons, each receiving signals from the ten FSR402 sensors positioned around the hand. The hidden layer is designed to balance accuracy with model efficiency, aiming to achieve high performance with minimal computational complexity. The output layer consists of a single neuron, which generates the predicted type of hand movement to be executed by the prosthesis. An example of the ANN model architecture is illustrated in Fig. 4.

For comparison with the ANN model, RNN-LSTM architecture is also designed. The optimal model, determined from the comparative evaluation of these two approaches, will be deployed in the hand prosthesis control system. The RNN-LSTM model is composed of four main components: the input layer, the LSTM layer, the fully connected layer, and the output layer. Unlike the ANN, the RNN-LSTM incorporates an additional LSTM layer, which consists of recurrently connected blocks known as memory blocks. Each block contains one or more memory cells+gating units—namely the input gate, output gate, and forget gate—which regulate the flow of information and enable long-term dependency learning. The deep learning models are implemented using the Python programming language and the TensorFlow library on the Google Colab platform, where training is accelerated through the use of Tensor Processing Units (TPUs). This significantly reduces training time compared to conventional devices. An example of the RNN-LSTM model architecture is illustrated in Fig. 5 [27].

IV. Results and Discussion

A. Results of Hand Prosthesis Control System Design

Based on the proposed design, the hand prosthesis control system is implemented using the InMoov open-source prosthesis model. The control system is responsible for coordinating multiple device subsystems to actuate finger movements. Its primary objective is to recognize pressure patterns generated by muscle activity during finger motion, using machine learning as the decision-making mechanism. In this implementation, the control system integrates a Raspberry Pi 4B, Arduino Mega 2560, and ESP32 as the main controllers. The FSR402 force sensors, connected to the Arduino Mega 2560, capture muscle pressure signals and provide input to the control system. Finger movements of the prosthesis are actuated by MG996R servo motors, which are driven by the ESP32 microcontroller. The overall control system is organized into three main modules: the Master Controller, the **S**lave Sensor, and the Slave Servo. The physical appearance of the hand prosthesis is shown in Fig. 6, while Fig. 7 presents the schematic diagram of the complete electronic circuitry.

The Master Controller serves as the primary processing unit of the control system, implemented using the Raspberry Pi 4B as both the data processor and decision-making component. It communicates with the Slave Sensor and Slave Servo modules via serial communication for data exchange. The machine learning model is executed on this device, enabling real-time interpretation of sensor inputs and generation of control commands. The Raspberry Pi 4B requires approximately 15 W of power for stable operation. When powered by a 12 V battery, a DC-DC step-down converter is employed to regulate and stabilize the output voltage to 6 V with a maximum current of 2.5 A. The schematic of the Master Controller circuit is presented in Fig. 8.

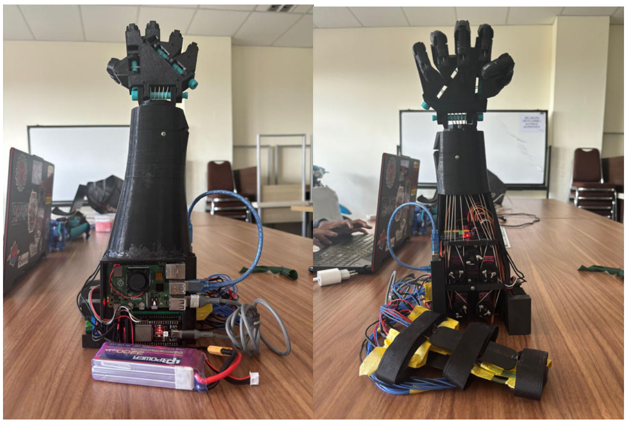

(a) (b)

Fig. 6. View of the hand prosthesis in (a) front view and; (b) rear view.

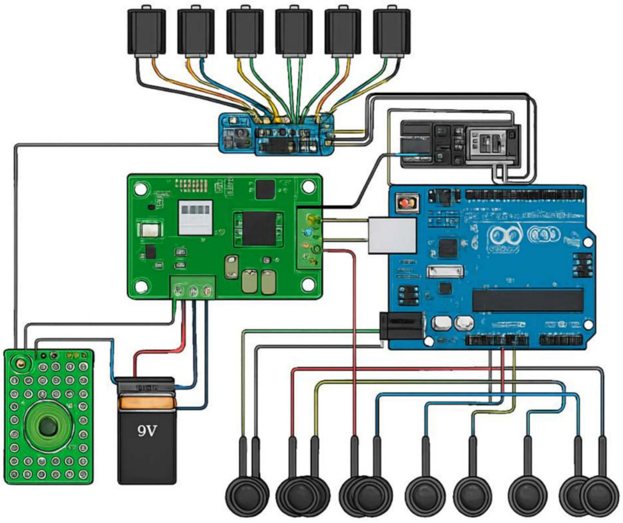


Fig. 7. Schematic of the hand prosthesis control system circuit.

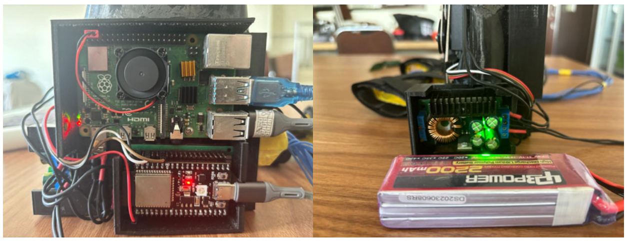

(a) (b)

Fig. 8. The master controller circuit consists of (a) a Raspberry Pi 4 Model B and (b) a power supply.

For decision-making, the Master Controller requires input data to be processed by the machine learning model. This data is acquired through the Slave Sensor circuit, which consists of an Arduino Mega 2560 microcontroller connected to multiple FSR402 force sensors. The circuit functions to capture and transmit pressure signals generated by the FSR402 sensors, serving as the primary input to the control system. The sensors are strategically positioned around the hand to detect muscle contractions associated with finger movements [29]. The force sensor reading system is executed on the Slave Sensor module. Data acquired from the FSR402 sensors is processed by the Arduino Mega 2560 and transmitted to the Master Controller via serial communication using a USB interface. The physical implementation of the Slave Sensor is shown in Fig. 9.

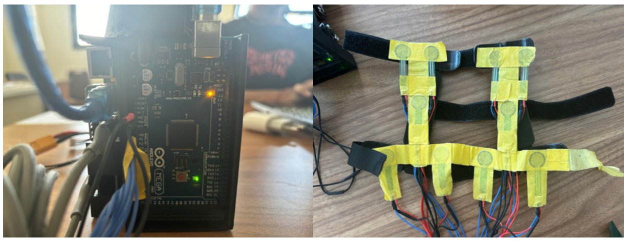

(a) (b)

Fig. 9. Slave sensor circuit consists of (a) Arduino mega 2560 and (b) FSR402 sensor socket.

The FSR402 force sensor has two connector terminals. For the ten sensors used in this design, the first terminal of each sensor is connected via blue cables to the analog input pins (A0–A9) of the Arduino Mega 2560, serving as the signal paths. The second terminal of each sensor is connected via black cables to a 10 kΩ resistor, which is then linked to the GND pin to complete the circuit.

The force sensor reading system is responsible for measuring the pressure values generated by muscle activity in the hand during finger movements. This system employs ten FSR402 sensors that are attached to the hand to capture these signals. The acquired data are represented as force values in Analog-to-Digital Conversion (ADC) units**,** with a measurable range from 0 to 1023. The collected data are then transmitted to the Master Controller, where they are processed by the deep learning neural network system for movement classification and decision-making.

B. Dataset Collection of Hand Prosthesis Control System

The dataset used in this study was obtained from the hand prosthesis. It consists of eleven columns: ten columns representing the values from the FSR402 force sensors and one column containing the corresponding data labels. Data collection is performed manually using the Slave Sensor module in combination with the Data Streamer feature in Microsoft Excel. This feature enables real-time recording of sensor readings and stores the data in .csv format, which is subsequently utilized for training the deep learning neural network system. The dataset collection process is illustrated in Fig. 10.

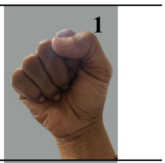
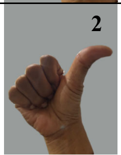
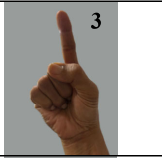
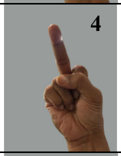
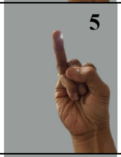
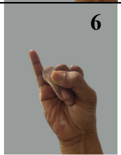
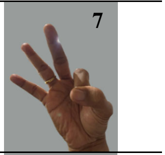
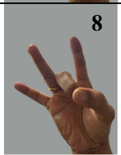
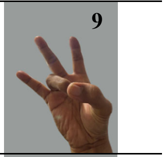
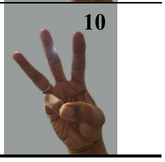
Fig. 10. FSR402 sensor dataset acquisition.

This study utilizes a dataset of 1000 samples, divided into training and testing subsets using an 80:20 ratio. The dataset is labeled according to ten distinct hand movements performed by the prosthesis: closed hand, thumb, index finger, middle finger, ring finger, little finger, thumb with index finger, thumb with middle finger, thumb with ring finger, and thumb with little finger. The dataset size of 1000 samples is relatively small for deep learning, which generally requires larger datasets to ensure robust generalization. In this study, the data were obtained from sensor measurements related to finger movement and prosthesis control, which are time-consuming and costly to collect. To overcome this limitation, we applied data augmentation, such as introducing controlled noise into the sensor signals, performing time-shifting, and segmenting longer recordings into smaller overlapping windows. These techniques effectively increased the diversity of training data without requiring additional experiments.

In addition, we used regularization techniques including dropout layers, early stopping, and weight decay to minimize overfitting and improve the model's stability during training. These strategies enabled the network to capture meaningful patterns from the limited dataset while maintaining good generalization performance. Nevertheless, we acknowledge that 1000 samples remain limited, and future work will focus on collecting larger datasets to further improve the robustness and reliability of the prosthesis control model.

The values captured by the FSR402 sensors are measured in ADC units, representing the digital form of the analog signals detected by the sensors. These ADC values can be converted into force values (Newtons) using calibration or an appropriate mathematical formula derived from the sensor's characteristics. Typically, a linear equation is applied to map the ADC range to the corresponding force values. Based on the specifications of the FSR402 sensor, the ADC output ranges from 0 to 1023, corresponding to a measurable force range of 0 to 100 N. Representative samples of the dataset for each movement label are presented in Table I.

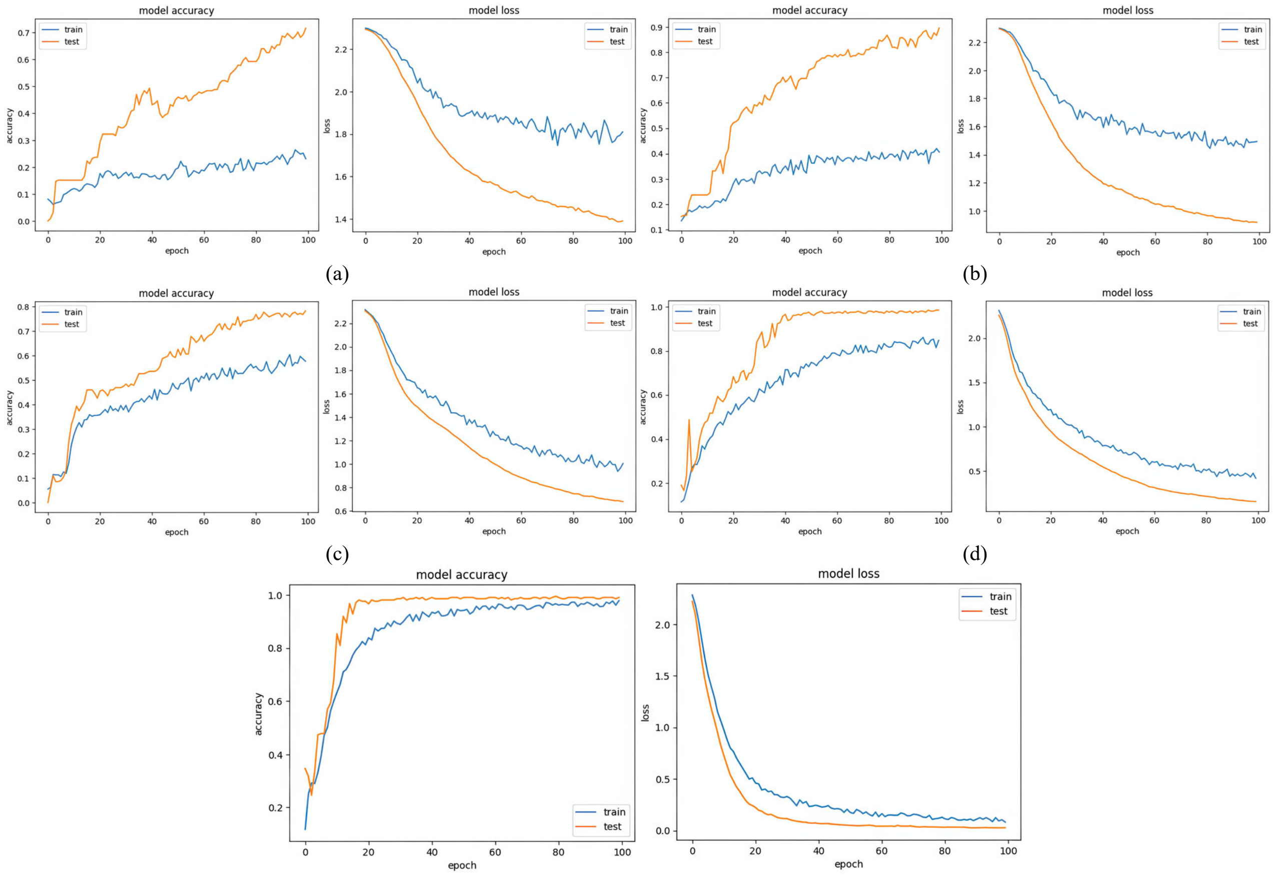
TABLE I. SAMPLE SENSOR DATA FROM EACH

F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	Label Data
0	0	0	442	502	0	0	0	335	94	1 
0	0	0	425	465	0	0	0	302	0	2 
0	0	0	87	254	0	0	0	147	0	3 
137	0	0	0	105	0	0	0	0	0	4 
292	0	0	379	278	0	0	0	226	0	5 
0	0	0	353	272	0	0	0	205	0	6 
412	0	86	0	0	0	0	0	0	0	7 
405	0	53	95	0	0	0	0	0	0	8 
450	120	32	0	0	0	39	0	0	0	9 
539	533	0	0	0	0	84	0	0	0	10 

In Table I, the values from the ten force sensors represent the initial readings recorded when the hand is moved according to each label. For Label 1, the active sensors are located at points 4, 5, 9, and 10. For Label 2, the active sensors are at points 4, 5, and 9. For Label 3, the active sensors are also at points 4, 5, and 9. For Label 4, the active sensors are at points 1 and 5. For Label 5, the active sensors are again at points 1 and 5. For Label 6, the active sensors are at points 4, 5, and 9. For Label 7, the active sensors are at points 1 and 3. For Label 8, the active sensors are at points 1, 3, and 4. For Label 9, the active sensors are at points 1, 2, 3, and 7.

Finally, for Label 10, the active sensors are at points 1, 2, and 7. It is important to note that for Labels 2, 3, and 6, the same sensor points (4, 5, and 9) are activated; however, the recorded sensor values at these points differ. This variation in sensor readings across similar activation patterns constitutes the input features that the machine learning model learns in order to accurately distinguish and predict the intended hand movement.

In the deep learning neural network system, two models will be created, namely ANN and RNN-LSTM using the Tensorflow library and the Google Colab platform.



(a)

(b)

(c)

(d)

(e)

Fig. 11. Training and testing results on the ANN model in the form of (a) accuracy and loss with 2 neurons, (b) accuracy and loss with 4 neurons, (c) accuracy and loss with 8 neurons, (d) accuracy and loss with 16 neurons, (e) accuracy and loss with 32 neurons.

As a result of executing the source code presented in Fig. 11, graphs were generated to illustrate the training and testing performance of both the ANN and RNN-LSTM models under several variations in the number of neurons in the hidden layer. These graphs depict the influence of the epoch parameter on model accuracy and loss values. An epoch represents the number of times the model utilizes the entire training dataset during the learning process. In the graphs, the training performance is represented by the blue line, while the testing performance is represented by the orange line.

Analysis of the model accuracy graph reveals a consistent upward trend as the number of epochs increases. During the initial training phase (epochs 0–30), the model exhibits a substantial improvement in accuracy, reflecting its ability to learn underlying patterns in the data and minimize prediction errors. However, as training progresses, the rate of improvement diminishes, indicating convergence toward an optimal solution. Prolonged training or the use of excessive epochs introduces the risk of overfitting, whereby the model becomes overly specialized to the training dataset, including its noise, thus reducing its ability to generalize effectively to unseen data during testing and validation. This phenomenon is evident in the RNN-LSTM model with 32 neurons, where the final testing and validation accuracy values are lower than the final training accuracy. A comparable trend is observed in the loss curve, though inversely related to accuracy: loss decreases sharply during early epochs, followed by a gradual reduction, and ultimately exhibits signs of overfitting with extended training duration.

For a more detailed comparison of the accuracy and loss of the ANN and RNN-LSTM models, the results of the training and testing processes are summarized in Table II.

Based on Table II, the model with two neurons achieved an accuracy of only 71.56% for the ANN and 34.09% for the RNN-LSTM, with relatively high loss values of 1.3895 and 1.6940, respectively. Such loss values, being greater than 1, indicate poor model performance and an inability to make reliable predictions. When the number of neurons in the hidden layer was increased to four, the loss values decreased to 0.9178 (ANN) and 0.9333 (RNN-LSTM), while the accuracy improved to 78.20% and 76.23%, respectively. However, these results still suggest limited predictive capability, as the loss values remain comparatively high. Further improvements in both loss reduction and accuracy were observed as the number of neurons increased to eight and sixteen. At 32 neurons, the ANN achieved a substantial decrease in loss to 0.0284, with accuracy rising from 99.05% to an even higher value, while the RNN-LSTM recorded a loss reduction to 0.0633 and an accuracy improvement to 97.21%. Overall, the results demonstrate that increasing the number of neurons generally leads to higher accuracy and lower loss in both models. Based on repeated experimental runs, statistical analysis of final testing accuracy values between ANN and RNN-LSTM demonstrated a statistically significant improvement ($p < 0.05$) for the RNN-LSTM model. This confirms that the observed performance gap in the accuracy and loss curves is not due to random variation but reflects a true performance difference.

TABLE II. DATA TRAINING AND TESTING MODEL

Model	Architecture			Training	Testing and Validation
	Number of Neurons	**Accuracy (%)**	**Loss**	**Accuracy (%)**	**Loss**
ANN	2	23.21	1.8091	71.56	1.3895
	4	40.71	1.4913	78.20	0.9178
	8	57.74	1.0036	89.57	0.6845
	16	84.76	0.4212	98.58	0.1562
	32	97.86	0.0837	99.05	0.0284
RNN-LSTM	2	24.72	1.9610	34.09	1.6940
	4	50.72	1.2817	76.23	0.9333
	8	68.32	0.7835	82.40	0.4504
	16	85.06	0.3968	92.00	0.2188
	32	97.65	0.0734	97.21	0.0633

C. Functionality of Hand Prosthesis

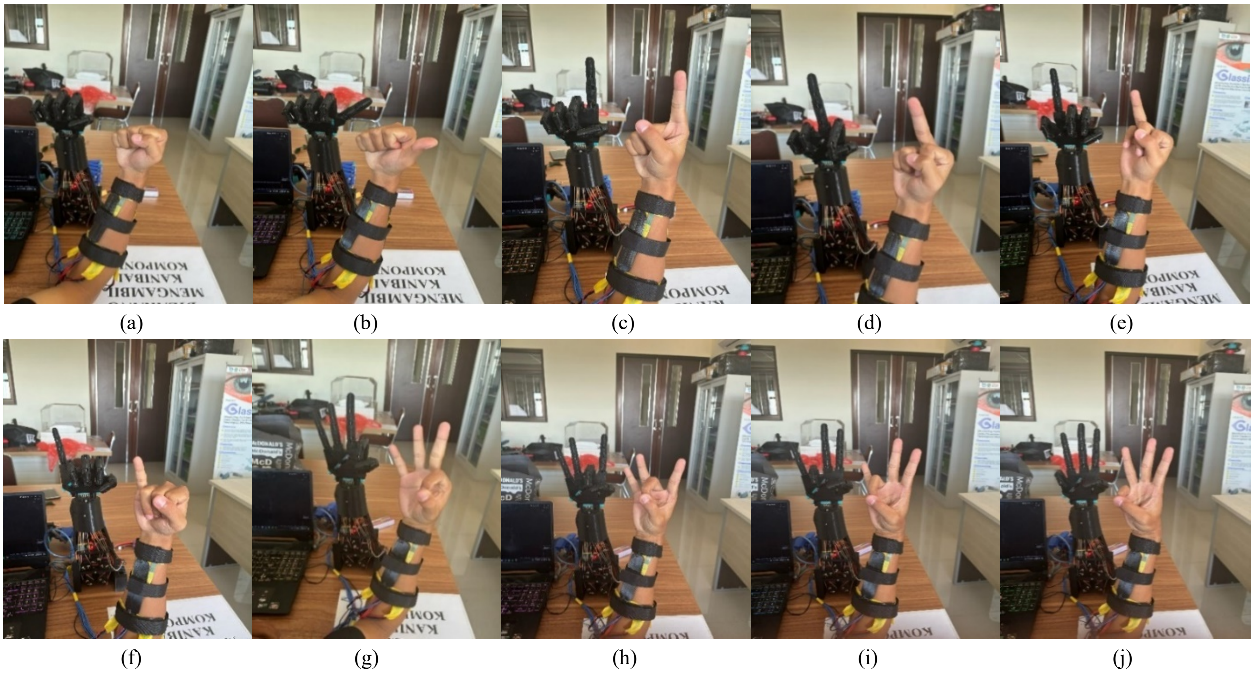
The testing of the hand prosthesis control system aims to evaluate both the functional performance of finger movements and the ability of the trained machine learning model to generate accurate predictions. The movements of the hand prosthesis are controlled by an array containing the PWM setpoint values for each servo in the Slave Servo code. This array defines three specific setpoints: 125, 300, and 400. These setpoints, expressed in PWM units, are converted into angular positions (degrees) based on two reference ranges: a PWM range of 125–575 and the corresponding servo angular range of 0°–180°.

Setpoint 125 corresponds to the initial finger position at 0°. At this setpoint, the servo rotates Clockwise (CW) up to 110°, producing a maximum finger joint movement of 90° when the finger is fully bent. At setpoint 300, the servo rotates Counterclockwise (CCW) by 70°, pulling the tendon string and resulting in approximately 45° of finger joint movement from the initial position. At setpoint 400, the servo rotates CCW by 110°, pulling the tendon string further and producing 90° of finger joint movement from the initial position. The servo movement configurations for each setpoint are summarized in Table III.

TABLE III. THE SERVO MOVEMENT CONFIGURATIONS

Label		Servo 1	Servo 2	Servo 3	Servo 4	Servo 5
Label 1	Setpoint	300	400	400	400	400
	Angle	70° CCW	110° CCW	110° CCW	110° CCW	110° CCW
Label 2	Setpoint	125	400	400	400	400
	Angle	110° CW	110° CCW	110° CCW	110° CCW	110° CCW
Label 3	Setpoint	300	125	400	400	400
	Angle	70° CCW	110° CW	110° CCW	110° CCW	110° CCW
Label 4	Setpoint	300	400	125	400	400
	Angle	70° CCW	110° CCW	110° CW	110° CCW	110° CCW
Label 5	Setpoint	300	400	400	125	400
	Angle	70° CCW	110° CCW	110° CCW	110° CW	110° CW
Label 6	Setpoint	300	400	400	400	125
	Angle	70° CCW	110° CCW	110° CW	110° CW	110° CW
Label 7	Setpoint	300	400	125	125	125
	Angle	70° CCW	110° CCW	110° CW	110° CW	110° CW
Label 8	Setpoint	300	125	400	125	125
	Angle	70° CCW	110° CW	110° CCW	110° CW	110° CW
Label 9	Setpoint	300	125	125	400	125
	Angle	70° CCW	110° CW	110° CW	110° CCW	110° CW
Label 10	Setpoint	300	125	125	125	400
	Angle	70° CCW	110° CW	110° CW	110° CW	110° CCW

Functional testing of the hand prosthesis is conducted for each label. The sensor socket is attached to the hand to measure the pressure exerted by the muscle on the sensor. The acquired sensor values are transmitted to the Master Controller, which immediately generates a prediction. The Master Controller terminal displays the received sensor data, prediction latency, probability distribution across the 10 labels, and the corresponding output executed by the Slave Servo.


(a) (b) (c) (d) (e)

(f) (g) (h) (i) (j)

Fig. 12. Display of a hand with a sensor socket attached as input to the control system at (a) Label 1, (b) Label 2, (c) Label 3, (d) Label 4, (e) Label 5, (f) Label 6, (g) Label 7, (h) Label 8, (i) Label 9, (j) Label 10 which will move the hand prosthesis.

Fig. 12 illustrates the hand movements with the attached sensor socket and their correspondence to the movements executed by the hand prosthesis. A label test is considered successful when the prosthesis replicates the natural hand movement; otherwise, it is deemed failed. In the motion test for (a) Label 1, the prosthesis movement matches the hand movement, and the same applies to the remaining labels (b–j), demonstrating consistent replication of hand movements by the prosthesis.

For Label 1, the active sensors are located at points 4, 5, 9, and 10, with a minimum prediction probability of 91.21% and output 0. For Label 2, the active sensors are at points 4, 5, and 9, with a minimum probability of 96.41% and output 1. For Label 3, the active sensors are at points 4, 5, and 9, with a minimum probability of 96.77% and output 2. For Label 4, sensors 1 and 5 are active, yielding a minimum probability of 98.80% and output 3. For Label 5, sensors 1 and 5 are active, with a minimum probability of 98.01% and output 4. For Label 6, sensors 4, 5, and 9 are active, with a minimum probability of 97.11% and output 5. For Label 7, sensors 1 and 3 are active, with a minimum probability of 92.97% and output 6. For Label 8, sensors 1, 3, and 4 are active, with a minimum probability of 93.88% and output 7. For Label 9, sensors 1, 2, 3, and 7 are active, with a minimum probability of 97.62% and output 8. Finally, for Label 10, sensors 1, 2, and 7 are active, with a minimum probability of 97.76% and output 9.

The summarized test results are presented in Table IV.

TABLE IV. HAND PROSTHESIS FUNCTIONALITY TEST DATA

Arm	Working Sensor	Arm Prosthesis			Success/Failure
		Prediction (%)	Output	Label	
Label 1	4, 5, 9, 10	91.21	0	Label 1	Success
Label 2	4, 5, 9	96.41	1	Label 2	Success
Label 3	4, 5, 9	96.77	2	Label 3	Success
Label 4	1,5	98.80	3	Label 4	Success
Label 5	1, 4, 5	98.01	4	Label 5	Success
Label 6	4, 5, 9	97.11	5	Label 6	Success
Label 7	1,3	92.97	6	Label 7	Success
Label 8	1, 3, 4	93.88	7	Label 8	Success
Label 9	1, 2, 3, 7	97.62	8	Label 9	Success
Label 10	1, 2, 7	97.76	9	Label 10	Success

Data in Table IV demonstrate that the integrated system operates effectively. The FSR402 sensor successfully measures the pressure exerted on the hand, as indicated by the sample sensor values, which are consistent with those recorded during testing. The correspondence between the generated outputs and the intended labels confirms that the deep learning neural network can accurately recognize muscle force patterns and make correct predictions.

V. Conclusion

The development of a machine learning—based control system for hand movement classification in a hand prosthesis has been successfully implemented, consisting of both hardware and software components. The hardware includes a Master Controller (Raspberry Pi 4B) for computation, a Slave Sensor (Arduino Mega 2560) with an FSR402 force sensor, and a Slave Servo module comprising an ESP32 microcontroller, a PCA9685 PWM servo driver, and an MG996R servo motor as the actuator. The software consists of a force-sensing system deployed on the Arduino Mega 2560 and a deep learning neural network implemented on the Raspberry Pi 4B, utilizing an ANN architecture with 32 neurons that achieved an accuracy of 99.05% and a loss of 0.0284.

Functional testing demonstrated that the integrated system operates reliably, with sensor activations during testing aligning with the training and testing data samples. The deployed model consistently produced prediction values exceeding 90% across all labels, and the prosthesis movements closely replicated the corresponding natural hand movements with the attached sensor socket. These findings indicate that the proposed model achieved accurate classification and reliable control across all ten predefined labels.

However, the integration of multiple microcontrollers and machine learning modules inevitably increases the architectural complexity and system cost, potentially constraining large-scale deployment and reducing accessibility for patients in resource-limited settings. Moreover, the current prototype employs wired communication between subsystems, which may introduce constraints on portability and signal latency. Future studies should therefore aim to streamline the control architecture by employing a unified high-performance embedded platform capable of executing both sensing and inference tasks in real time. Emphasis should also be placed on enhancing energy efficiency and identifying cost-effective hardware components without compromising classification accuracy or mechanical responsiveness. In addition, expanding the training dataset and conducting validation with amputee participants will be essential to improve model generalization, robustness, and clinical applicability.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

IWW served as the conceptor, writer, and designer of the hand robot prosthesis; IMW and IMEDP developed the control system design; IGPAS was responsible for the design and assembly of the robot; DNKPN conducted the literature study with input from partners; and TGTN analyzed the research results; all authors had approved the final version.

Funding

This research on the robotic prosthesis was funded through a grant from the Udayana University Research and Community Service Institute (LPPM) under a research implementation assignment agreement (SP3) financed by PNBP funds for the 2024 fiscal year, with contract number B/255.496/UN14.4.A/PT.01.03/2024.

Acknowledgment

We would like to express our sincere gratitude to the Udayana University Research and Community Service Institute (LPPM) for providing PNBP research funding, and to the Mechatronics Laboratory team for their dedicated support and hard work.

References

[1] I. A. Satam, "Review studying of the latest development of prosthetic limbs technologies," *Int. J. Sci. Eng. Res.*, vol. 12, no. 12, pp. 721–731, 2021.

[2] P. S. Mckechnie and A. John, "Anxiety and depression following traumatic limb amputation: A systematic review," *Injury*, vol. 45, no. 12, pp. 1859–1866, 2014. doi: 10.1016/j.injury.2014.09.015

[3] K. S. K. Prasad, I. Fauzi, I. M. E. D. Adiputra *et al.*, "Optimizing the performance of an arm prosthesis control system using the PID tuning method with MATLAB," *Maj. Ilm. Teknol. Elektro*, vol. 22, no. 2, 165, 2023. doi: 10.24843/mite.2023.v22i02.p02 (in Indonesian)

[4] D. T. Hutchinson, "The quest for the bionic arm," *J. Am. Acad. Orthop. Surg.*, vol. 22, no. 6, pp. 346–351, 2014. doi: 10.5435/JAAOS-22-06-346

[5] W. Daly, L. Voo, T. Rosenbaum-Chou *et al.*, "Socket pressure and discomfort in upper-limb prostheses: A preliminary study," *J. Prosthetics Orthot.*, vol. 26, no. 2, pp. 99–106, 2014. doi: 10.1097/JPO.0000000000000021

[6] W. Widhiada, T. G. T. Nindhia, and I. N. Budiarsa, "Design of the bionic robot hand with electromyography smart sensor based on MATLAB/Simulink," *Int. J. Mech. Eng. Robot. Res.*, vol. 9, no. 8, pp. 1122–1127, 2020. doi: 10.18178/ijmerr.9.8.1122-1127

[7] R. B. Azhiri, M. Esmaeili, and M. Nourani, "EMG-Based feature extraction and classification for prosthetic hand control," *Epic Ser. Comput.*, vol. 83, pp. 136–145, 2022. doi: 10.29007/zflb

[8] H. K. P. Y. Jayaweera. *Design and Implementation of Electromyography (EMG) Based Real-Time Pattern recognition Model for Prosthetic Hand Control*, UK: Liverpool John Moores University, 2021.

[9] Z. Chen, H. Min, D. Wang *et al.*, "A review of myoelectric control for prosthetic hand manipulation," *Biomimetics*, vol. 8, no. 3, pp. 1–22, 2023. doi: 10.3390/biomimetics8030328

[10] L. I. B. López, F. M. Ferri, J. Zea *et al.*, "CNN-LSTM and post-processing for EMG-based hand gesture recognition," *Intell. Syst. with Appl.*, vol. 22, 200352, 2024. doi: 10.1016/j.iswa.2024.200352

[11] Z. Wang, W. Huang, Z. Qi *et al.*, "MS-CLSTM: Myoelectric manipulator gesture recognition based on multi-scale feature fusion CNN-LSTM network," *Biomimetics*, vol. 9, no. 12, 784, 2024. doi: 10.3390/biomimetics9120784

[12] H. E. Williams, A. W. Shehata, K. Y. Cheng *et al.*, "Myoelectric prosthesis control using recurrent convolutional neural network

regression mitigates the limb position effect," *IEEE Transactions on Biomedical Engineering*, vol. 69, no. 7, pp. 2243–2255, 2024. doi: 10.1109/TBME.2022.3140269

[13] D. C. Joshi, P. Kumar, R. C. Joshi *et al.,* "AI-enhanced analysis to investigate the feasibility of EMG signals for prosthetic hand force control incorporating anthropometric measures," *Prosthesis*, vol. 6, no. 6, pp. 1459–1478, 2024. doi: 10.3390/prosthesis6060106

[14] I. M. E. D. AdiPutra, K. K. S. Prasad, I. Fauzi *et al.,* "Evaluation of a hand motion classification machine learning model for a hand prosthesis prototype control system," *Maj. Ilm. Teknol. Elektro*, vol. 22, no. 1, 141, 2023. doi: 10.24843/mite.2023.v22i01.p18 (in Indonesian)

[15] M. Barfi, H. Karami, F. Faridi *et al.,* "Heliyon improving robotic hand control via adaptive fuzzy-PI controller using classification of EMG signals," *Heliyon*, vol. 8, no. 12, e11931, 2022. doi: 10.1016/j.heliyon.2022.e11931

[16] V. Gohel and N. Mehendale, "Review on electromyography signal acquisition and processing," *Biophysical reviews*, vol. 12, no. 6, pp. 1361–1367, 2020.

[17] A. Jan, I. Ul, M. Anas *et al.,* "Development and evaluation of a novel Force Myography (FMG) based human-machine interface for transradial prosthetics: A comparative study with electromyography (EMG) techniques," *Mehran University Research Journal of Engineering & Technology,* vol. 43, no. 4, pp. 55–62, 2024.

[18] S. Huang and H. Wu, "Texture recognition based on perception data from a bionic tactile sensor," *Sensors*, vol. 21, no. 15, 5224, 2021.

[19] V. Marco, G. Massimo, and G. Manuel, "Additive manufacturing of flexible Thermoplastic Polyurethane (TPU): Enhancing the material elongation through process optimisation," *Prog. Addit. Manuf.*, vol. 10, no. 4, pp. 2877–2891, 2025. doi: 10.1007/s40964-024-00790-y

[20] S. Surya and S. Ramamoorthy, "Assistive device to control prosthetic hand movements using machine learning approach," *International Journal of Health Sciences*, vol. 6, no. S1, pp. 1386–1396, 2022.

[21] A. Belford, S. A. Moshizi, A. Razmjou *et al.,* "Using miniaturized strain sensors to provide a sense of touch in a humanoid robotic arm," *Frontiers in Mechanical Engineering*, vol. 6, 550328, 2020. doi: 10.3389/fmech.2020.550328

[22] B. Taşar, A. Gülten, and O. Yakut, "Design and manufacturing of 15 DOF myoelectric controlled prosthetic hand," *Pamukkale University Journal of Engineering Sciences,* vol. 26, no. 5, pp. 884–892, 2020. doi: 10.5505/pajes.2020.12268

[23] H. Bayoumi, M. I. Awad, and S. A. Maged, "An improved approach for grasp force sensing and control of upper limb soft robotic prosthetics," *Micromachines*, vol. 14, no. 3, 2023. doi: 10.3390/mi14030596

[24] I. W. Widhiada, I. G. B. W. Kusuma, A. A. G. P. Diputra *et al.,* "Integrated control system of leg prostheses based on deep learning neural network," in *AIP Conf. Proc*., 2024, vol. 3053, no. 1, 020001. doi: 10.1063/5.0199791

[25] J. Gantenbein, C. Ahmadizadeh, O. Heeb *et al.,* "Feasibility of force myography for the direct control of an assistive robotic hand orthosis in non - impaired individuals," *J. Neuroeng. Rehabil.,* vol. 8, pp. 1–13, 2023. doi: 10.1186/s12984-023-01222-8

[26] A. Y. Prathama, A. Aminullah, and A. Saputra, "ANN (Artificial Neural Network) approach for determining the percentage of job weight and estimating the value of structural work in primary hospitals," *J. Teknosains,* vol. 7, no. 1, pp. 14–25, 2018. doi: 10.22146/teknosains.30139 (in Indonesian)

[27] A. Toro-Ossaba, J. Jaramillo-Tigreros, J. C. Tejada *et al.,* "LSTM recurrent neural network for hand gesture recognition using EMG Signals," *Applied Sciences*, vol. 12, no. 19, pp. 1–21, 2022. https://doi.org/10.3390/app12199700

[28] W. Yuan, C. Liu, F. Liu *et al.,* "Learning-based predictive beamforming for UAV communications with jittering," *IEEE Wirel. Commun. Lett.,* vol. 9, no. 11, pp. 1970–1974, 2020. doi: 10.1109/LWC.2020.3009951

[29] E. C. Swanson, E. J. Weathersby, J. C. Cagle *et al.,* "Evaluation of force sensing resistors for the measurement of interface pressures in lower limb prosthetics," *J. Biomech. Eng.*, vol. 141, no. 10, pp. 1–46, 2019. doi: 10.1115/1.4043561