

RESEARCH

Open Access



Surface EMG driven gesture recognition using machine learning for robotic applications

Iva Valentinova Valcheva¹, Vaibhav Gandhi^{2*} and Eris Chinellato¹

*Correspondence:

Vaibhav Gandhi

V.Gandhi@soton.ac.uk

¹Design Engineering and Mathematics, Middlesex University, London, UK

²University of Southampton, Delhi Campus, Gurgaon, India

Abstract

This paper explores non-invasive use of surface electromyographic (sEMG) signals from a human arm for controlling various devices. In recent years, numerous studies have explored sEMG-based gesture recognition for prosthetic and robotic applications. Gesture sets are often limited to a small number of movements, restricting the range of control. Furthermore, the translation of offline classification results to real-time robotic control remains challenging due to latency, signal variability, and computational overhead. These limitations motivate further research into robust, adaptable, and computationally efficient sEMG-based control systems. An 8-sensor Myo Armband device is employed for sEMG signal acquisition. This study involves four participants—comprising an equal number of men and women with diverse ages and body conditions. Each participant performed nine different gestures, repeated 10 times, yielding a comprehensive training dataset. Various machine learning algorithms were applied to filter the raw signals, scale the data and classify the gestures using optimized parameters. During evaluation, the most effective filtering methods and classifiers (with subject-specific tuning) were selected for near-real-time gesture classification and robotic device control. The trained model predicts the performed gesture from the nine available classes and transmits the corresponding command to the robotic system. The chance or baseline accuracy of the system thus translates to 11.1% with the probability of randomly selecting one correct gesture out of nine. The evaluation phase demonstrated that Random Forest, Linear SVM, and Extra Trees were the top three classifiers. The classification model achieved an average accuracy of 92.8%. Despite the promising results, real-time classification for robotic control remains a challenge, necessitating further refinement of gesture segmentation and signal processing techniques.

1 Introduction

Electromyography (EMG) is a technique used to diagnose or evaluate the electrical signals produced in the skeletal muscles [1]. These muscles are attached to bones and tendons and are responsible for the movement of the skeleton. There are various types of skeletal muscles throughout the body, consisting of bundles of fibres that contain proteins called myosin and actin, which work together to produce muscular contractions [2]. There are two main types of electromyography: surface electromyography (sEMG)



© The Author(s) 2026. **Open Access** This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by-nc-nd/4.0/>.

and intramuscular electromyography (iEMG). The sEMG is a non-invasive technique that involves placing electrodes on the skin over the area of interest to acquire the muscle signals. This approach is practical and easy to use but has its limitations, such as signal interference caused by factors such as the amount of fat surrounding the muscles or the distance between the muscles and electrodes. In contrast, the iEMG is an invasive technique that involves inserting needle electrodes directly into the muscles. This method is more precise than sEMG, as it allows for the detection of individual motor unit activity, providing more detailed information about the neuromuscular function. However, specialized equipment and trained professionals are required to manage it. Both techniques have their advantages and disadvantages, and the choice between the two depends on the specific requirement. Of recent, there has been a growing interest in using EMG signals for controlling devices and robots [3–6]. While conventional interfaces such as joypads offer reliable and low-effort control of robotic platforms, EMG-based control provides a more natural and intuitive human–robot interaction modality. This approach can be particularly advantageous in assistive or rehabilitation contexts, or in environments where manual control is impractical. This paper focuses on using sEMG signals from a Myo Armband and machine learning techniques to classify hand gestures for robotic control.

This paper is organized as follows. Section II gives an overview of previous studies on EMG-based control mechanisms. Section III discusses recent techniques for pre-processing, feature extraction, and classification of EMG signals, some of which have been implemented within the proposed work. Section IV presents the results of the proposed approaches. Section V concludes the paper.

2 Review of EMG-based control mechanisms

EMG has been investigated and applied in areas such as robotics, exoskeletons, and prosthetics, among others, to enhance the functionality and improve the lives of individuals [3–11]. Samuel et al. [8] proposed an intelligent EMG pattern recognition model for upper-limb multifunctional prostheses. The project aimed to create a prosthetic limb that could function as a natural limb by detecting muscle signals and translating them into the desired action. In the development of exoskeletons, Cenit and Gandhi [3] designed and developed an sEMG-based exoskeleton strength enhancer for the legs. The exoskeleton was designed to help people with mobility challenges. The system utilized EMG signals to detect when the leg muscle required support and translated this information into a desired mechanical action, thus allowing the user to manoeuvre more efficiently. Additionally, Antonius and Tjahyadi [12] designed an electromyography gesture identification system using a convolutional recurrent neural network for controlling quadcopters. The project aimed to create a more efficient control system for quadcopters by detecting hand gestures. Likewise, the development of low-cost, wireless, 3-D-printed custom armbands for sEMG hand gesture recognition by Côté-Allard et al. [9] aimed to create an affordable system for individuals with mobility challenges. The armband was designed to detect the user's hand movements using EMG signals and translate them into the desired action. The system proved to be accurate and effective in detecting hand movements. Furthermore, DelPreto et al. [10] developed a plug-and-play supervisory control system using muscle and brain signals for real-time gesture and error detection. The system aimed to enhance the control of robotic systems by detecting the user's

movements and errors. Several studies have also focused on signal processing and classification techniques for EMG signals. Chowdhury et al. [1] reviewed different signal processing techniques and classification methods for EMG signals. The study explored different techniques such as wavelet transform, independent component analysis, and artificial neural networks, among others. The review concluded that machine learning techniques had better classification accuracy than traditional techniques. Another study by Khezri and Jahed [11] proposed a real-time intelligent pattern recognition algorithm for surface EMG signals utilizing linear discriminant analysis (LDA) and support vector machine (SVM) to classify the EMG signals accurately. Furthermore, Gandhi et al. [6] proposed a novel quantum neural network-based surface EMG signal filtering approach for the control of a robotic hand. The study aimed to filter the EMG signals to remove noise. In addition to the technical challenges, ethical considerations are also important in the development of EMG-based control methods. Another important aspect to consider is also the privacy and security of the user's EMG data. As these technologies become more pervasive, it is important to ensure that user data is protected and not misused. Overall, the papers and projects reviewed here demonstrated the exciting potential for a wide range of EMG-based applications. From controlling mobile robots and exoskeletons to improving prosthetic limb functionality and even enabling hybrid brain-robot interfaces, the possibilities for EMG-based control systems are immense [13–15]. However, there are also significant technical and ethical challenges that must be addressed as these technologies continue to develop and become more widespread. EMG-based systems are inherently sensitive to factors such as electrode placement, signal variability, and user-specific calibration, which must be carefully managed to ensure safety, repeatability, and robustness in real-time applications.

While recent studies have explored advanced learning architectures such as CNNs, quantum neural models, and hybrid EMG–EEG control systems, these approaches often require substantial computational resources and large-scale datasets, limiting their feasibility for real-time deployment on embedded robotic platforms. In contrast, classical techniques such as Linear SVM and Random Forest remain attractive for their interpretability, low latency, and adaptability to smaller datasets. Moreover, many previous studies focused on either signal processing or offline classification performance, without integrating the complete pipeline of data acquisition, processing, classification, and robotic actuation in a single ROS-based system [16]. This work addresses these gaps by presenting a low-cost framework that combines robust preprocessing, feature optimization, and lightweight classifiers for robotic control through EMG signals. This approach prioritises computational efficiency and generalizability across users, contributing to the development of a practical EMG-controlled system.

3 Development and analysis of results

This section details the development process involved in acquiring the EMG signals using a Myo Armband¹ and controlling an end device. The acquired EMG data is saved, labelled with appropriate tags, and scaled to ensure uniformity. The raw EMG signals are also filtered and subsequently fed to a machine learning algorithm for classification. An accurate classification model is saved to be used in the future for real-time classification.

¹The Myo Band is a wearable armband with electromyographic (sEMG) sensors and an inertial measurement unit (IMU) that detects muscle activity and motion for gesture-based control of devices.

During real-time operation, the classified class (performed gesture) is sent as a message to execute an associated robotic task.

4 Data acquisition

4.1 Sensors

Myo armband (cf. Figure 1), made by Thalmic Lab., is used to acquire the sEMG signals. The band is equipped with 8 sEMG sensors for detecting electrical activity through muscular movements in a range of 0–2000 μV . The sensors are powered by a rechargeable Lithium-ion battery, which can last up to 8 h, and have a range of use of up to 10 m.

Proper placement and calibration of the Myo band is crucial for accurate EMG signal acquisition. To properly position the Myo armband, slide it up your arm and centre the rectangular sensor pad on the muscle. Ensure that the Myo logo is facing up, and the sensors are in contact with the skin. Calibration ensures that the device is accurately detecting and interpreting the EMG signals.

4.2 Forearm muscles

Once the Myo armband is placed on the forearm, the sensors can specifically obtain the EMG signals from the Posterior and Anterior muscles [17] (cf. Figures 2 and 3). Flexor Carpi Radialis muscle is located on the medial side (inner side) of the forearm and is responsible for flexing the wrist. Extensor Carpi Radialis muscle is located on the lateral side (outer side) of the forearm and is responsible for extending the wrist. Brachioradialis muscle is located on the lateral side of the forearm, closer to the elbow, and is responsible for flexing the elbow. Pronator Teres muscle is located on the medial side of the forearm, closer to the elbow, and is responsible for pronating the forearm (turning it so the palm faces downward). Extensor Digitorum muscle is located on the lateral side of the forearm and is responsible for extending the fingers. Flexor Digitorum muscle is located on the medial side of the forearm and is responsible for flexing the fingers.



Fig. 1 Myo Armband

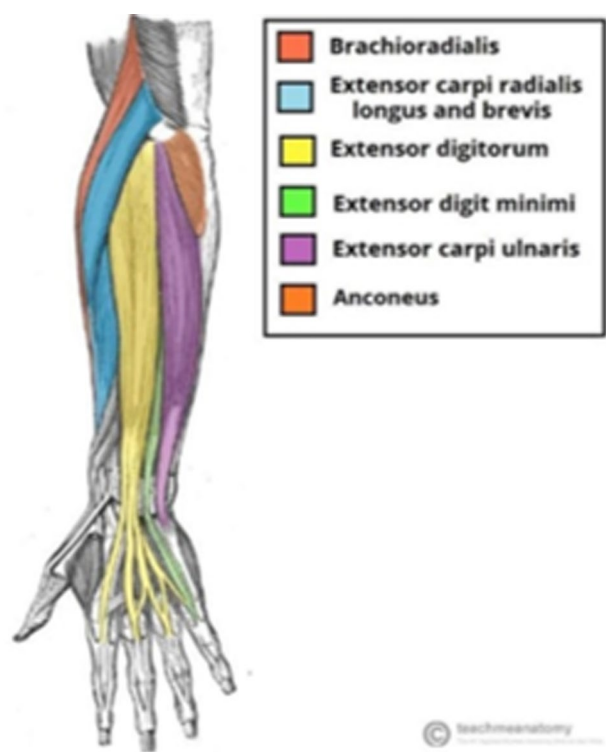


Fig. 2 Posterior Muscles

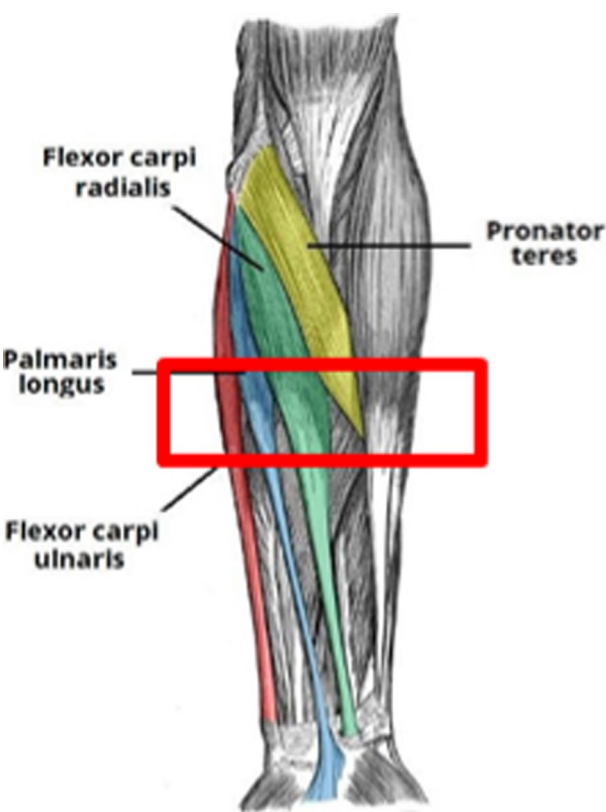


Fig. 3 Anterior Muscles

4.3 Gesture selection

Appropriate gestures are chosen for data recording, based on the location of the armband and the muscles that produce the most relevant EMG. The red rectangle (cf. Figure 3) represents the armband location.

The selection of appropriate gestures is critical for accurate data collection and subsequent classifier training. Based on these criteria, the chosen gestures are detailed below and shown in (Figs. 4, 5, 6, 7, 8, 9, 10, 11, 12).

Switch: Press the tip of your thumb against the base of your middle finger, then quickly release the tension by flicking your middle finger downward to create a snapping sound (two times) (Fig. 4).

Forwards: To make a gun shape with one hand, point your index finger straight out and curl your middle, ring, and little fingers towards your palm (Fig. 5).

Backwards: hold your hand with the palm facing up and fingers slightly apart, then curl your fingers towards your palm and repeat the motion a few times, beckoning the person towards you (Fig. 6).

Right: Same as forwards but pointing to the right. (Fig. 7) **Up:** Same as forwards but pointing up with the wrist (Fig. 8).

Down: Same as forwards but pointing down with the wrist (Figs. 9).

Freeze: Twist your wrist 180 degrees to the right, like unscrewing a lightbulb (Fig. 10).

Left: Same as forwards but pointing to the left. (Fig. 11) **On/Off:** closing your hand making a fist shape and hitting it against the other hand; second, open the hand and hit it against your other hand as how you did with the fist; finally, after hitting with the open hand, keep the position and twist the hand 90 degrees to the right (Fig. 12).

By carefully selecting the appropriate gestures, one can ensure that the data collected is separable, relevant and reliable. This, in turn, can facilitate accurate classification of EMG signals and ultimately contribute to the development of more effective human–robot interaction systems. This is an important step towards advancing our

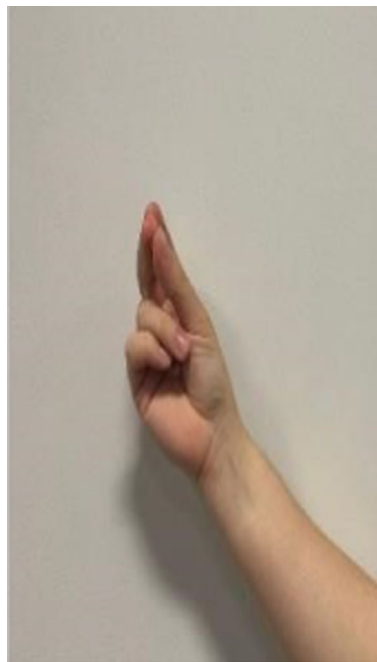


Fig. 4 Switch

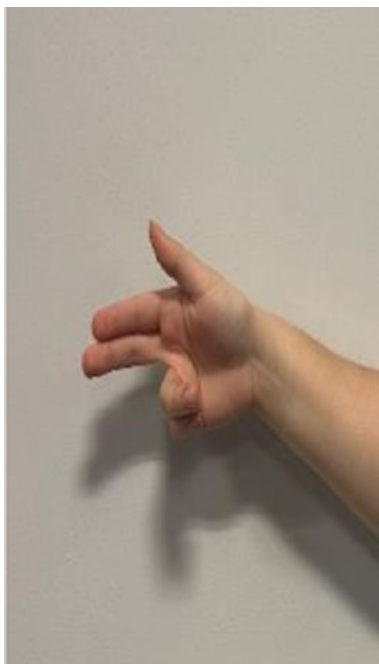


Fig. 5 Forwards

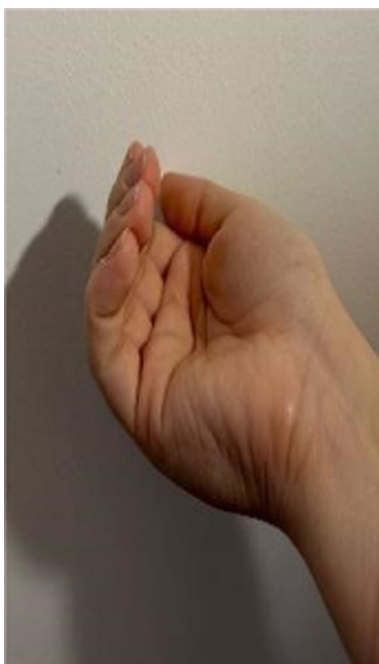


Fig. 6 Backwards

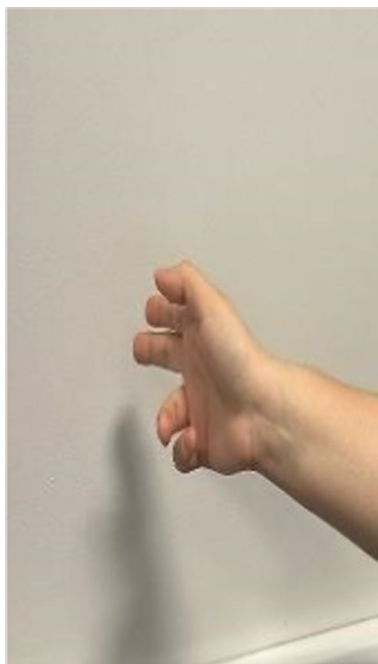


Fig. 7 Right

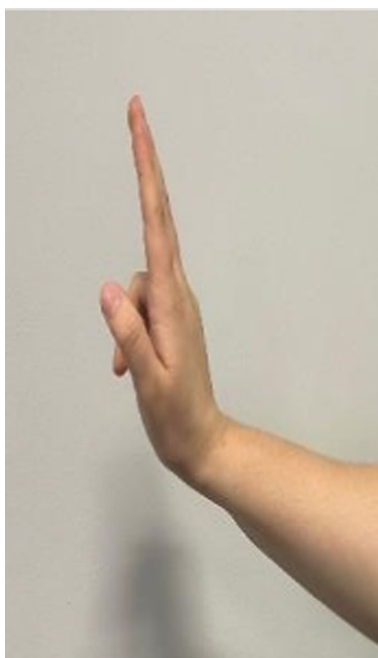


Fig. 8 Up

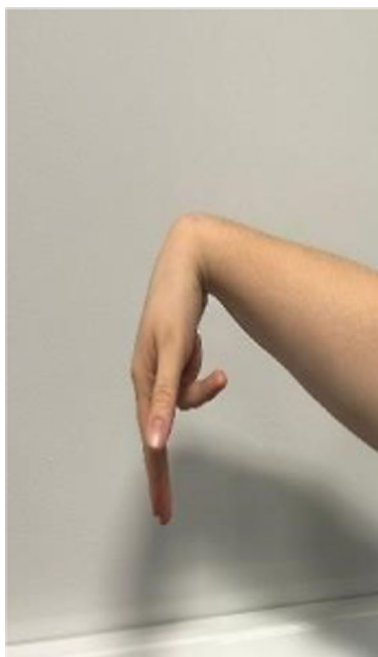


Fig. 9 Down



Fig. 10 Freeze



Fig. 11 Left



Fig. 12 On/Off (Combination of 3 gestures)

understanding of the role that gestures can play in EMG signal classification, and the significance of this research for future development in this field.

4.4 Signal pre-processing

Filtering, smoothing, and normalization are fundamental steps in any physiological signal processing. EMG signals can be subject to various types of noise, including electrical interference, physiological signals from adjacent muscles that can contaminate the signal and obscure relevant information as well as movement artefacts amongst others. Thus, filtering is essential to eliminate unwanted signals and improve the quality of the EMG signal.

To ensure that only gesture-relevant frequencies are utilized while unwanted components are suppressed, a band-pass filter of 4th order was applied. The cut-off frequencies were determined by analysing the gesture samples, where a preliminary spectral assessment indicated that the dominant frequency components for the gestures were concentrated below 50 Hz. The recorded EMG amplitude ranged approximately from $-80 \mu\text{V}$ to

80 μV , with the precise frequency content varying across gestures (cf. Figure 13). Based on this analysis, a 20–50 Hz band-pass filter was chosen to preserve the relevant components while attenuating frequencies below 20 Hz which are also typically associated with baseline drift and motion artefacts, and those above 50 Hz which are often linked to sensor noise and high-frequency interference. Although this narrower band excludes some higher-frequency EMG information, it was found to be sufficient for accurate gesture classification in the present application.

Signal smoothing is also utilized to reduce the variability of EMG signals. Moving average and Butterworth filters are among the methods that can be applied to smooth out the signal. To remove unwanted frequencies outside the selected cut-off frequencies, a bandpass filter of 4th order was utilized. This enabled a sharper cut-off and a steeper roll-off than lower-order filters and ensured that unwanted frequencies are attenuated effectively. After applying the bandpass filter, the signals were further smoothed using forward–backward filtering, a technique that ensures zero-phase distortion by applying the filter in both directions. This effectively increases the filter order. Subsequently, a low-pass filter was applied to further smooth the signal and extract relevant features for analysis. The combination of the bandpass filter and forward–backward filtering technique helps to improve the signal quality and reduce unwanted noise, which is essential for accurate signal classification.

Rectification is another important commonly used signal processing technique in EMG signal analysis (cf. Figure 14). This allows the EMG signal to be analysed based on its amplitude while removing the negative component of the signal; essentially converting the signal from bipolar to unipolar for improving feature extraction and

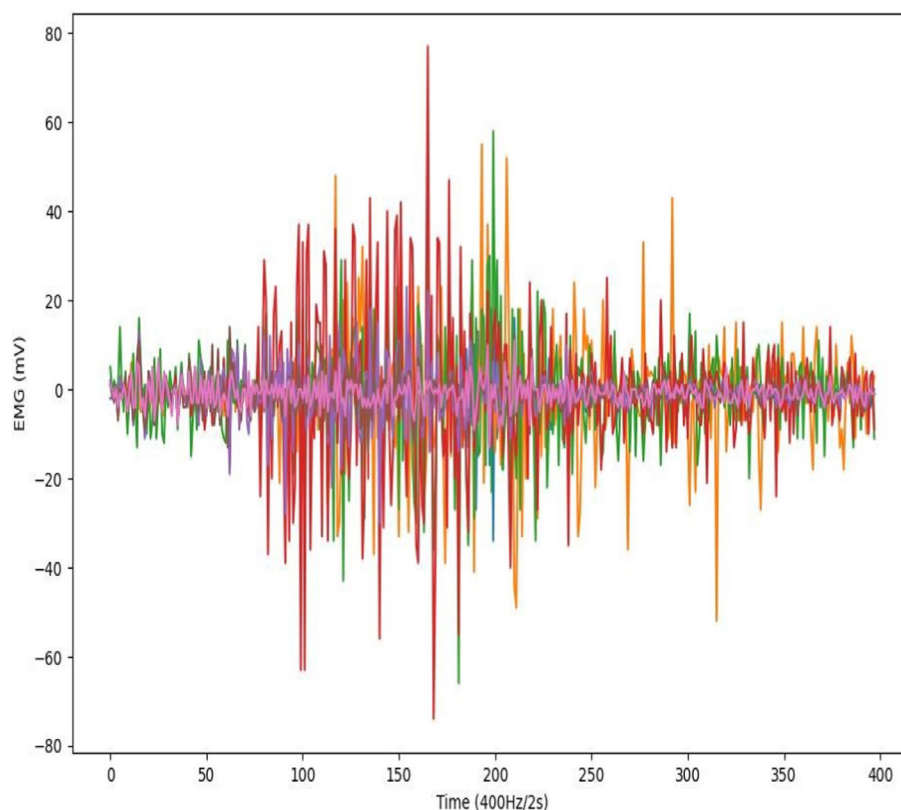


Fig. 13 Raw EMG signals for one gesture

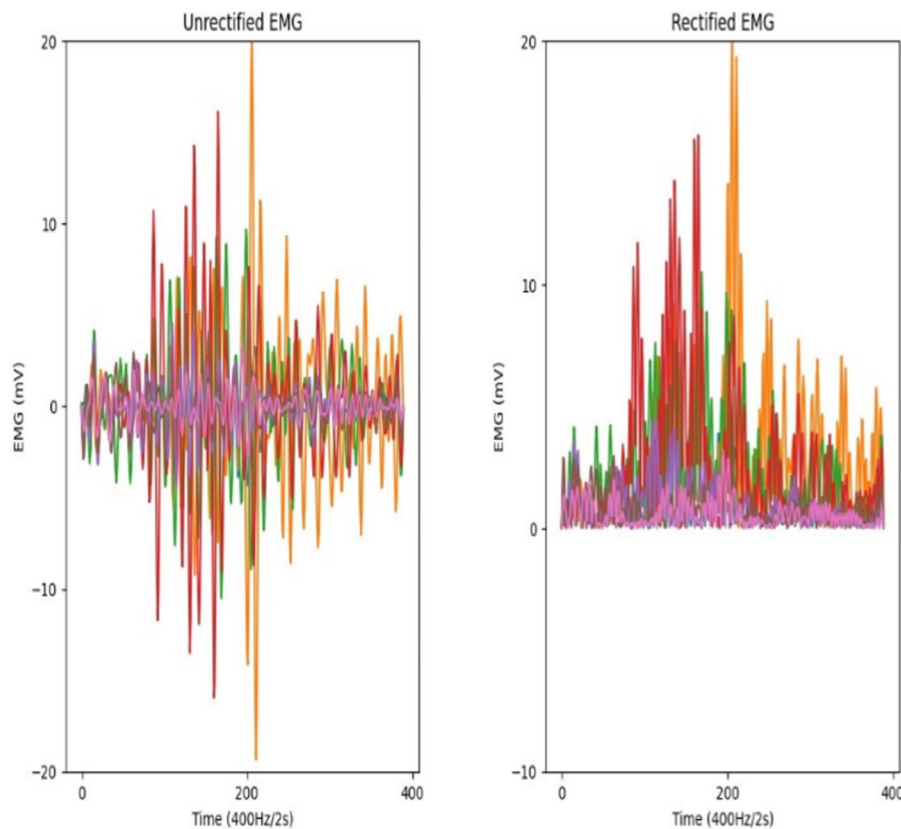


Fig. 14 Difference between an unrectified and a rectified signal

classification. It is also common to normalize the EMG signal to ensure consistency, comparability and improve processing accuracy. This involves scaling the signal values to a standard range before applying filtering techniques. If referring to the filter's cut-off frequencies, normalization means expressing them as a fraction of the sampling frequency to ensure proper filter design and performance across different sampling rates. This allows for easy comparison and analysis of filter designs, independent of the actual sampling frequency. By normalizing the cut-off frequencies to the sampling frequency, filters can be easily transferred across different applications and sampling frequencies, making them more versatile and flexible. The normalization process allows for consistent and accurate analysis of the frequency response of a filter, making it an important step in digital signal processing.

After applying the filtering steps, the signal is much clearer and easier to analyse. As illustrated in Fig. 15, the difference between the original and filtered signal is noticeable, demonstrating that the signal is now ready to be used for classification purposes. The filtered signal has a better signal-to-noise ratio, making it more reliable for extracting features and performing subsequent processing. The improvement in the signal quality achieved through filtering is critical in ensuring accurate classification of the gestures and ultimately improving the overall performance of the system.

4.5 Feature extraction

There are various feature extraction techniques for EMG signals that are suitable for real-time classification. However, the optimal technique selection depends on specific

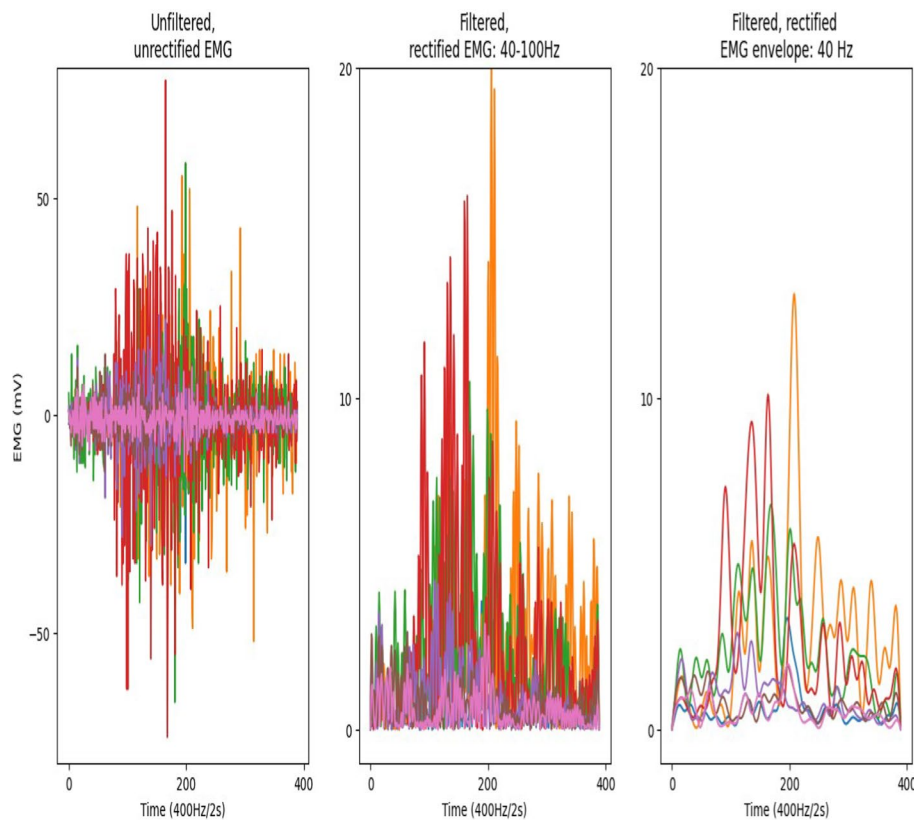


Fig. 15 Raw and Clean signal

application requirements. In this study, different feature extraction techniques such as mean absolute value (MAV), waveform length (WL), mean absolute value slope (MAVS), number of crossings of mean (NCM), difference absolute mean value (DAMV), simple Square Integral (SSI), root mean square (RMS), variance of absolute values (VAR), absolute differential signal (ADS) and zero crossing (ZC) before and after filtering the signal were investigated (cf. Table 1).

Tables 2 and 3 present the results before and after filtering, respectively. The post-filtering analysis exhibited almost identical results with a mean accuracy of 87.56%, with the highest accuracy of 100% and never below 75% over five sets. On the other hand, some features as Zero Crossing (ZC), Number of Crossings of Mean (NCM), and Simple Square Integral (SSI). Showed very low accuracy, even 0%, in the pre-filtered signals while some others as Mean Absolute Value Slope (MAVS), Difference Absolute Mean Value (DAMV), and Absolute Differential Signal (ADS) demonstrated 100% accuracy.

The most optimal features in this case are MAV, VAR, MAVS and RMS. To ensure consistent feature extraction results, it is crucial to validate code correctness, data consistency, and random seeds settings. Combining or averaging results from multiple runs can also improve the robustness of feature values estimation.

4.6 Classifier

To improve the identification of EMG signals, a study was conducted using multiple classifiers to determine the most accurate one for real-time applications, ensuring its effectiveness in controlling robotic devices. Table 4 represents comparison of the different

Table 1 Extracted EMG features and their mathematical definitions

Feature	Mathematical Expression	Description
MAV	$(1/N) \sum_{i=1}^N x_i $	Mean absolute value
WL	$\sum_{i=1}^{N-1} x_{i+1} - x_i $	Waveform length
MAVS	$(MAV_2 - MAV_1) / \Delta t$	Mean absolute value slope (rate of change of MAV across two consecutive windows)
NCM	$\# \{i \in [1, N - 1] : (x_i - \bar{x})(x_{i+1} - \bar{x}) < 0\}$	A typical implementation of number of times the signal crosses its mean value counts indices where consecutive samples lie on opposite sides of the mean $\bar{x}$
DAWV	$(1/(N - 1)) \sum_{i=1}^{(N-1)} x_{(i+1)} - x_i $	Difference absolute mean value (aka the mean absolute first difference)
SSI	$\sum_{i=1}^N x_i^2$	Simple square integral (total signal energy within the window)
RMS	$\sqrt{\frac{1}{N} \sum_{i=1}^N x_i^2}$	Root mean square (effective amplitude; related to muscle force)
VAR	$VAR = (1/(N - 1)) \sum_{i=1}^N (x_i - \bar{x})^2$	Variability of the signal around the mean
ADS	$ADS = \sum_{i=1}^{N-1} x_{(i+1)} - x_i $	Common useful variants are either the plain sum of absolute differences or the simple sum of differences

classifiers that were trained for each subject, both before and after filtering the signals, including when using mixed-subject datasets. The primary objective of including mixed subjects in the study was to evaluate the performance of the classifiers in real-world situations where multiple users may operate the same device. The outcomes derived from these were used to establish the best classifier that can subsequently be used for controlling robotic devices in near-real-time. Table 5 summarizes the accuracy of different feature extraction methods after filtering. These results collectively highlight the effectiveness of various classifiers in distinguishing EMG signals under different conditions, and enabled selecting the most appropriate classifier. The 3 classifiers with high classification accuracy were Random Forest, Linear SVM and Extra Trees. A confusion matrix was also obtained subsequently (cf. Figure 16).

Based on these results, the Linear Support Vector Machine (SVM) was identified as the optimal classifier for distinguishing gestures in near-real-time. The model performed well using its default settings, with the initial trial conducted using SVC (kernel = “linear”, $C = 0.025$).

4.7 Robot operating system (ROS)

ROS is an open-source framework used for developing robot software that works with a variety of programming languages but more prominently with C++, Python and MATLAB. To evaluate the control capability of the Myo armband after filtering and classifying the EMG signals, it was used to operate a TurtleBot4 Lite via the Turtlesim program. The TurtleBot 4 Lite is a mobile robot that is equipped with many sensors and can be controlled through ROS. In this project, the TurtleBot 4 Lite is controlled using a socket connection. To accomplish the goal, a Python script is run on the TurtleBot 4 Lite. The script creates a socket server and listens the data sent by the client and converts it into commands for the robot to execute. The Python script creates a ROS Node, in this case “turtle_controller”. The Node is responsible for publishing messages to the velocity topic (/turtle1/cmd_vel), which the TurtleBot 4 Lite listens to for movement commands. The Node subscribes to the socket connection and reads incoming data, which it then translates into commands for the robot. The socket server is created using the socket module, which allows for socket connections to be created, accepted, and closed. The server is set to listen on IP address and the provided port. When a connection is accepted, a new thread is created to handle incoming data from the client for incoming connections on port 3020. When a client connects to the socket server, the TurtleBot 4 Lite receives the data sent by the client and converts it into commands for the robot to execute. The Python script creates a ROS Node, in this case “turtle_controller”. The Node is responsible for publishing messages to the velocity topic (/turtle1/cmd_vel), which the TurtleBot 4 Lite listens to for movement commands. The Node subscribes to the socket connection and reads incoming data, which it then translates into commands for the robot. The socket server is created using the socket module, which allows for socket connections to be created, accepted, and closed. The server is set to listen on IP address and the provided port. When a connection is accepted, a new thread is created to handle incoming data from the client.

When data is received from the client, it is decoded from bytes to a UTF-8 string, and the individual commands are extracted and added to a list of directions. The list of directions is processed every second by a timer call back function that pops the next

Table 2 Pre-filtering

Run	Classification score (s)	Accuracy (%)
1	86.91506457	87.5
2	19.77827281	87.5
3	34.34406432	87.5
4	27.54681759	87.5
5	24.98745324	87.5
6	43.79663307	87.5
7	120.8919654	100
8	46.13479829	93.75
9	118.7762123	100
10	29.20796394	87.5
11	116.2177027	93.75
12	130.3487888	93.75
13	18.5951972	0
14	24.97059222	87.5
15	52.82621723	93.75
16	23.79241705	0

Table 3 Post-filtering

Run	Classification score (s)	Accuracy (%)
1	86.91506457387691	100.0
2	19.778272807223775	87.5
3	34.34406431853859	87.5
4	27.546817586521865	87.5
5	24.987453239631	87.5
6	43.79663306523119	87.5
7	120.89196541732309	100.0
8	46.13479829436573	93.75
9	118.77621227873885	100.0
10	29.207963939431816	87.5
11	156.82794150425615	93.75
12	105.96827431604886	100.0
13	23.79241705362272	0.0
14	152.7747322703375	87.5
15	134.2487992555653	100.0

direction from the list and publishes it to the velocity topic. The TurtleBot 4 Lite can move forward, backward, turning left, and turning right. The Python script translates the incoming commands into movement commands by assigning appropriate values to the linear and angular velocities of the Twist message. The message is then published to the "/turtle1/cmd_vel" topic.

4.8 Controlling the Turtlebot

Turtlebot 4 Lite is used to demonstrate the gesture classification functionality. However, the device has certain limitations that prevents some desired functions from being implemented. The robot can move along two axes: it can rotate about the Z-axis and translate linearly along the X-axis. It cannot move along the Y-axis; therefore, to move right or left, it must first rotate about the Z-axis. For e.g., if it is required to turn 90 degrees to the left, it is necessary to rotate the robot first in Z axis and then move in the X axis. Additionally, the robot has some default safety measures to prevent damage which restricts it from making negative movements. As a result, it can only move in

Table 4 Classifier performance (including mixed-subjects)

No	Classifiers	Subject 1		Subject 2		Subject 3		Subject 4		2 mixed subjects		4 mixed subjects	
		Pre-filtering	Post-filtering	Pre-filtering	Post-filtering	Pre-filtering	Post-filtering	Pre-filtering	Post-filtering	Pre-filtering	Post-filtering	Pre-filtering	Post-filtering
0	Nearest_neighbors	0.111111	0.833333	0.111111	0.888889	0.166667	0.888889	0.166667	0.777778	0.138889	0.916667	0.097222	0.791667
1	Linear_SVM	0.111111	0.944444	0.111111	0.944444	0.166667	0.888889	0.166667	0.888889	0.111111	0.972222	0.166667	0.777778
2	Polynomial_SVM	0.055556	0.277778	0.055556	0.055556	0.055556	0.333333	0.055556	0.166667	0.055556	0.333333	0.041667	0.277778
3	RBF_SVM	0.055556	0.055556	0.055556	0.055556	0.055556	0.055556	0.055556	0.055556	0.055556	0.055556	0.041667	0.041667
4	Gaussian_process	0.055556	0.055556	0.055556	0.055556	0.055556	0.055556	0.055556	0.055556	0.194444	0.194444	0.111111	0.111111
5	Gradient_boosting	0.111111	0.833333	0.277778	0.611111	0.222222	0.777778	0.388889	0.777778	0.277778	0.611111	0.291667	0.625000
6	Decision_tree	0.222222	0.777778	0.333333	0.555556	0.222222	0.888889	0.166667	0.611111	0.083333	0.555556	0.097222	0.541667
7	Extra_trees	0.277778	0.944444	0.277778	0.944444	0.444444	0.888889	0.333333	0.833333	0.305556	0.861111	0.180556	0.777778
8	Random_forest	0.444444	0.944444	0.444444	1	0.388889	0.944444	0.444444	0.944444	0.333333	0.916667	0.291667	0.847222
9	Neural_net	0.111111	0.555556	0	0.888889	0.111111	0.888889	0.111111	0.888889	0.083333	0.777778	0.111111	0.736111
10	AdaBoost	0.055556	0.388889	0.166667	0.166667	0.111111	0.222222	0.111111	0.222222	0.166667	0.111111	0.138889	0.319444
11	Naive_bayes	0.611111	0.722222	0.777778	1	0.888889	0.944444	0.611111	0.777778	0.777778	0.833333	0.541667	0.625000
12	QDA	0.055556	0.166667	0.111111	0	0.222222	0.111111	0.055556	0.111111	0.138889	0.166667	0.125	0.194444
13	SGD	0.055556	0.777778	0.166667	0.888889	0.055556	0.833333	0.166667	0.777778	0.111111	0.833333	0.138889	0.944444

the positive X axis. Another limitation is that multiple Turtlebots cannot have the same IP address. This creates a problem when trying to control multiple devices. For socket connections, the exact same IP address is required. Even if multiple Turtlebots are connected in this way, they cannot be controlled separately by assigning tasks to each one, instead, they would all perform the same task.

Overall, these limitations need to be considered when working with the Turtlebot 4 Lite. While it has some restrictions, it remains a useful device for demonstrating gesture-based classification control and can be used effectively. To overcome these limitations, Turtlesim has been used as a substitute. Specifically, Turtlesim has been used for tasks that the Turtlebot 4 cannot perform, such as reverse movements, movements in the Y axis, and control of multiple Turtlesim. Turtlesim has proven to be a useful alternative for these tasks and has helped to expand the range of functions that can be demonstrated in the project.

It is crucial to consider the latency between the EMG signal acquisition, processing, and subsequent command generation for the robot. In this case, the average time taken for the classifier to generate a command and send it to the Turtlebot is less than a millisecond, precisely 0.0009560585021972656 s), indicating that the system is highly responsive. It is important to note that the accuracy of the classifier is directly proportional to the effectiveness of the control commands sent to the robot. Thus, with accurate classification of the EMG signals, the robot will be able to quickly and accurately respond to user commands.

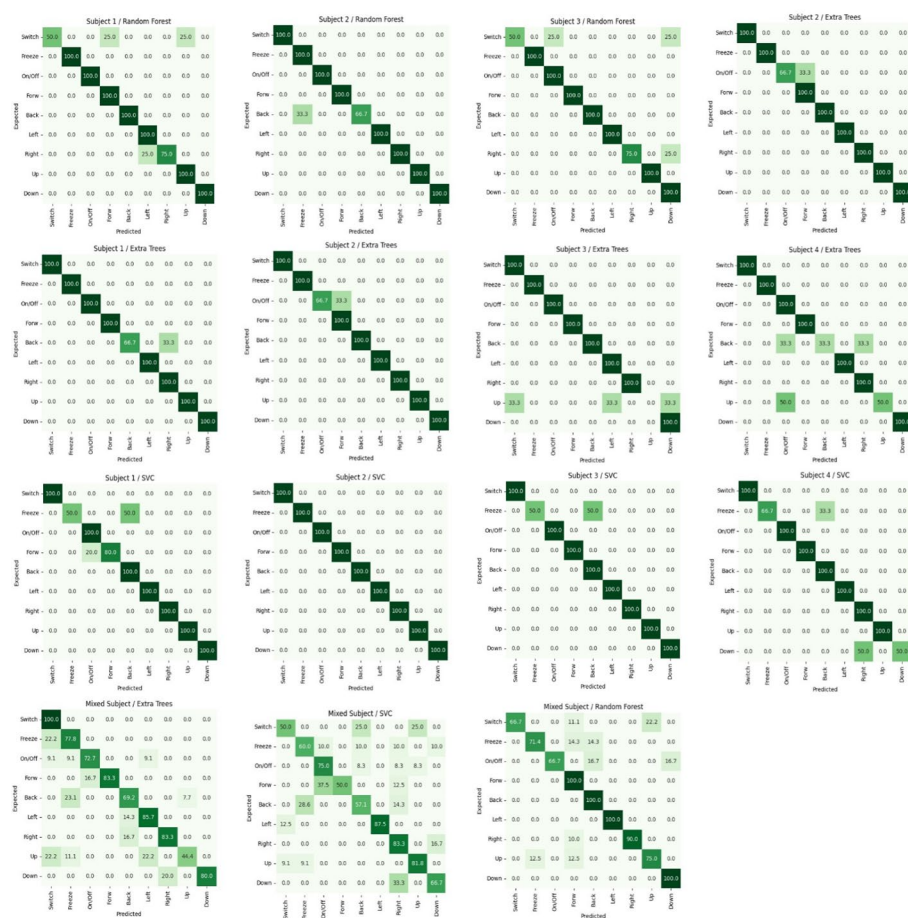
5 Insights

Based on the analysis of the results, it is determined that Random Forest is the best classifier among the top three, with an average accuracy of 92.8%, followed by Linear SVM with an average accuracy of 89.6%, and Extra Trees with an average accuracy of 87%. However, after examining the confusion matrices, and ultimately selected Linear SVM as the preferred classifier due to its lower number of prediction errors. The Linear SVM is trained with the initial set of gestures and achieved excellent results, with accuracy ranging from 88 to 100%. As for feature extraction, it was found that not all types of features are suitable after fine-tuning the filter parameters. The results show that some standard filter parameters yield better results compared to other types of feature extraction (Table 5).

Regarding robot control, the system successfully interprets and executes commands by implementing a communication protocol that receives messages from the gesture classifier. This ensures accurate timing and command execution. No separate test or evaluation dataset was used, as the model is intended for direct deployment in robot control tasks. While this limits the ability to report conventional generalization metrics, the training results provide an initial assessment of model behavior and performance. While the gesture classification achieved high accuracy using various feature extraction techniques, and the TurtleBot effectively responded to specific commands, challenges remain. The real-time classifier does not consistently predict all performed gestures with complete accuracy, highlighting the need for further improvements in classification robustness and real-time processing. Future work will need to focus on assessing inter-session robustness to enhance the model's generalizability and practical applicability in real-world robotic control scenarios.

Table 5 Classification with feature extraction post filter tuning

Feature	Performance	Interpretation
MAV (Mean absolute value)	94.4%	Strong performance—stable amplitude-based feature
RMS (Root mean square)	94.4%	Consistent with MAV; also measures signal power
VAR (Variance)	94.4%	Similar strength: amplitude-energy related
MAV_SLOPE	100%	Excellent—may capture temporal dynamics effectively
ZC (Zero crossings)	11.1%	Very poor—probably too sensitive to noise or not discriminative enough
WL (Waveform length)	16.6%	Weak—not contributing much
DAVM (Difference absolute value mean)	100%	Excellent—robust dynamic feature
NMC (Number of mean crossings)	11.1%	Poor—like ZC
SSI (Simple square integral)	5.55%	Extremely poor—possibly redundant with RMS/VAR
ADS (Average difference signal)	100%	Excellent—high discriminative power

**Fig. 16** Confusion matrices for individual & mixed-subject EMG classification results using different machine learning models

The development of a universal classifier for EMG signal recognition remains challenging due to the wide range of signal filtering techniques and the lack of a single model that consistently performs well across different subjects and conditions. EMG signals vary significantly based on factors such as electrode placement, muscle type, and movement characteristics, making it difficult for a single classifier to achieve high accuracy

in all scenarios. Consequently, researchers must experiment with multiple models and optimize them for specific applications.

The results of this study indicate that the primary challenge stems from the unique characteristics of each EMG signal and the variability in execution time. Although a fixed window of 2 s was used for signal processing at a sampling frequency of 200 Hz; gesture-specific variations in signal rise and fall times introduce inconsistencies in the recorded data. This variation can lead to the inclusion of extraneous signals, such as those generated when the arm is in a relaxed state, potentially reducing classification accuracy. Addressing this issue requires further refinement of gesture segmentation and signal processing techniques to improve model performance. However, most class-discriminative information for gross gestures such as hand open/close, wrist pronation/supination, etc. is carried in low-to-mid frequency bands and in time-domain patterns, so a 200 Hz sampling frequency suffices.

One potential approach to improving classifier performance is to use techniques such as adaptive thresholding and wavelet-based signal processing, which are commonly employed to detect peak EMG activity. These methods enable precise segmentation, enhancing real-time classification and improving control for prosthetic and other robotic systems.

Several studies have explored EMG-based gesture recognition using diverse feature extraction and classification methods. For example, Cai et al. [18] achieved good accuracy using an SVM-based classifier on upper-limb EMG signals, while Côté-Allard et al. [19] reported this using CNN-based models using custom EMG armbands. The proposed system, achieving an average accuracy of 92.8% with subject-specific tuning, is therefore comparable to these results while maintaining significantly lower computational complexity, making it more suitable for real-time robotic control. Deep learning approaches, such as the convolutional recurrent neural network model proposed by Antonius and Tjahyadi [12], have also demonstrated strong performance (>95%) but often require high computational resources and large datasets. In contrast, our approach relies on lightweight, interpretable machine learning models (e.g., Random Forest, Linear SVM), which balance accuracy and real-time feasibility.

Furthermore, unlike prior works focusing on offline classification or static gestures, our system integrates gesture classification with ROS-based robotic control, demonstrating near-real-time performance with negligible perceptual delay (<300 ms). This represents a practical step toward deploying EMG-controlled assistive systems.

6 Conclusion and future work

The proposed EMG-controlled mechanism project has shown promising results in terms of implementing a complete ROS based robot control system thorough EMG signals. Using a Turtlebot and an EMG armband, the proposed model recognized and responded to specific gestures. Our approach involved the use of several feature extraction techniques and machine learning algorithms, with the Linear Support Vector Classification (SVC) classifier ultimately providing the best results. However, despite the success of the project, there are still several challenges that need to be addressed in order to improve its accuracy and reliability. One of the main challenges is the variability of EMG signals, which can be affected by factors such as electrode placement, muscle type, and movement type. As a result, there is a need for more robust and adaptable algorithms that can

address these variations and accurately classify a wider range of EMG signals. Another challenge is the real-time processing of EMG signals which is computationally expensive. To reduce the delay in gesture recognition, the system must focus on optimizing signal processing and classification rather than using deep learning in real-time. Instead of fixed time segments, adaptive windowing can be implemented to adjust processing dynamically based on signal activity, ensuring only relevant information is used. Selecting fewer but more effective features, such as MAV and RMS can reduce computational load without compromising accuracy. Switching to computationally lesser intensive classifiers such as Decision Trees or Linear SVM can further speed up prediction compared to more computationally intensive and complex models. Running the system on optimized hardware, such as edge devices or faster processors, will further reduce response time and make gesture recognition faster and more efficient. A detailed latency breakdown across the acquisition–processing–command pipeline was not recorded in this study. Although only classifier latency was measured, the system exhibited no perceptible delay during operation, suggesting total latency within the acceptable range for real-time control (typically < 200–300 ms) [20]. The system is designed to operate in a near-real-time manner, meaning that the signal acquisition, feature extraction, and classification are performed within short processing windows suitable for responsive gesture recognition that enable rapid execution. While exact latency measurements were not within the scope of this study, the computational complexity of the pipeline is sufficiently low to ensure a low computational overhead, making the approach feasible for near-real-time or embedded implementations. However, future work should include a comprehensive latency analysis.

In addition, there is a need for further testing and validation of the proposed system in different scenarios and with a wider range of users. This would help to identify any limitations or issues with the system and provide insights into how it can be improved. The TurtleBot experiments in this study were intended as a proof-of-concept demonstration to validate the feasibility of the proposed EMG-based control framework in a real-time robotic environment. A comprehensive quantitative evaluation, including multiple repetitions per gesture and statistical analysis of classification accuracy, was beyond the current scope. The potential applications of an EMG-controlled mechanism system are vast and could have significant implications in various fields, including healthcare, rehabilitation, and robotics. Overall, the proposed EMG-controlled mechanism represents an important step towards the development of practical and effective EMG-based control systems. With further research and development, it has the potential to make a significant impact and improve the quality of life for many people.

Acknowledgement

For the purpose of open access, the author(s) has applied a Creative Commons attribution (CC BY) licence (where permitted by UKRI), 'Open Government Licence' or 'Creative Commons attribution no-derivatives (CC BY-ND) licence' may be stated instead) to any Author Accepted Manuscript version arising.

Author contributions

IV wrote the main manuscript text with corresponding figures and tables. VG and EC supervised, reviewed and edited the manuscript.

Funding

No funding was received for this work.

Data availability

The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

Declarations

Ethics approval and consent to participate

This study was approved by the Middlesex University's Ethical Approval Committee. All participants were adults. Given the minimal-risk nature of the study and the anonymised data collection procedures, written informed consent was not required. Informed consent was obtained from all participants, and consent to participate was implied by their voluntary completion of the study.

Consent for publication

Consent for publication is given.

Competing interests

The authors declare no competing interests.

Received: 13 September 2025 / Accepted: 2 February 2026

Published online: 19 February 2026

References

1. Chowdhury RH, et al. Surface electromyography signal processing and classification techniques. *Sensors*. 2013;13(9):12431–66.
2. Azhiri RB, Esmaeili M, Nourani M. Emg-based feature extraction and classification for prosthetic hand control. *arXiv*. 2021. <https://doi.org/10.48550/arXiv.2107.00733>.
3. Cenit M, Gandhi V. Design and development of the sEMG-based exoskeleton strength enhancer for the legs. *Journal of Mechatronics, Electrical Power, and Vehicular Technology*. 2020;10(2):61–71.
4. Bisi S, et al. Development of an EMG-controlled mobile robot. *Robotics*. 2018;7(3):36.
5. V. Krasin, V. Gandhi and Z. Yang, "EMG based elbow joint powered exoskeleton for biceps brachii strength augmentation," in July 12–17, 2015.
6. Gandhi V, McGinnity TM. Quantum neural network based surface EMG signal filtering for control of robotic hand. *IEEE Int Joint Conf Neural Net*. 2013. <https://doi.org/10.1109/IJCNN.2013.6706781>.
7. Abbaspour S, et al. Evaluation of surface EMG-based recognition algorithms for decoding hand movements. *Med Biol Eng Comput*. 2020;58:83–100.
8. Samuel OW, et al. Intelligent EMG pattern recognition control method for upper-limb multifunctional prostheses: advances, current challenges, and future prospects. *IEEE Access*. 2019;7:10150–65.
9. Côté-Allard U, et al. A low-cost, wireless, 3-D-printed custom armband for sEMG hand gesture recognition. *Sensors*. 2019;19(12):2811.
10. DelPreto J, et al. Plug-and-play supervisory control using muscle and brain signals for real-time gesture and error detection. *Auton Robot*. 2020;44:1303–22.
11. Khezri M, Jahed M. Real-time intelligent pattern recognition algorithm for surface EMG signals. *Biomed Eng Online*. 2007;6:1–12.
12. Antonius R, Tjahyadi H. "Electromyography gesture identification using CNN-RNN neural network for controlling quadcopters. *J Phy*. 2021. <https://doi.org/10.1088/1742-6596/1858/1/012075>.
13. Naser H, Hashim HA. sEMG-based hand gestures classification using a semi-supervised multi-layer neural networks with autoencoder. *Syst Soft Comp*. 2024;6:200144.
14. Rezaee K, et al. Hand gestures classification of sEMG signals based on BiLSTM-metaheuristic optimization and hybrid U-Net-MobileNetV2 encoder architecture. *Sci Rep*. 2024;14(1):31257.
15. Zafar MH, Langâs EF, Sanfilippo F. Empowering human-robot interaction using sEMG sensor: hybrid deep learning model for accurate hand gesture recognition. *Res Eng*. 2023;20:101639.
16. M. Krawczyk, V. Gandhi and Z. Yang, "Robot Operating System (ROS) Controlled Anthropomorphic Robot Hand," 2022.
17. TeachMeAnatomy. *Muscles of the posterior compartment of the forearm*. Available: <https://teachmeanatomy.info/upper-limb/muscles/posterior-forearm/>.
18. Cai S, et al. SVM-based classification of sEMG signals for upper-limb self-rehabilitation training. *Front Neurobot*. 2019;13:31.
19. U. Côté-Allard et al, "Transfer learning for sEMG hand gestures recognition using convolutional neural networks," in 2017 *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2017.
20. Scheme E, Englehart K. Electromyogram pattern recognition for control of powered upper-limb prostheses: state of the art and challenges for clinical use. *J Rehabil Res Dev*. 2011. <https://doi.org/10.1682/jrdd.2010.09.0177>.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.