# EMG Based HCI Device to Support Computer Operation

**Firman Isma Serdana[a,] *, Akif Rahmatillah[b], Soegianto Soelistiono[b]**
[a]Department of Electronics Engineering, Politeknik Elektronika Negeri Surabaya
PENS Campus, Jl. Raya ITS, Keputih, Sukolilo, Surabaya City, East Java 60111
[b]Departemen of Physics, Science and Technology Faculty, Universitas Airlangga
UNAIR C Campus, Mulyorejo, Kec. Mulyorejo, Kota SBY, Jawa Timur 60115
*E-mail: firmanisma@pens.ac.id

**Abstract**

The human computer interface is a method of interaction between a person and a computer that utilizes a human interface device. One example of this is using hand movements to control a computer pointer, which produces a unique electromyography signal for each basic movement direction (up, down, right, and left). This project utilized an artificial neural network with a structure of seven inputs, ten hidden layer nodes, and four outputs to classify electromyography signals from the brachioradialis and flexor carpum ulnaris muscles into four basic movement categories within an Arduino Uno. The artificial neural network was trained offline using a high-capacity machine for efficiency since the Arduino Uno has low raw processing capability. The Sparkfun Pro Micro's HID function and the mouse.move() library were used to translate the classification results into pointer movement on a PC. The classification rate for the prerequisites setting resulted in an average of 93.7375%, with individual classification rates of 96.55% for up movement, 93.4% for down movement, 90.85% for right movement, and 91.95% for left movement.

**Keywords:** artifical neural network; electromyography; human computer interface; human interface device

## 1. Introduction

The muscle is an active movement tool in the human body's musculoskeletal system. The muscle has the ability to move passive movement tools in the form of bones or a framework that is adjusted by the placement of the muscle on the skeletal system and the joint system that supports the direction of movement that can be performed by the muscle. These movement capabilities include rotational movements, two-way movements, four-way movements, and so on [1]. In every muscle activity, especially movement activity, the muscle is influenced and produces bioelectric activity that spreads from nerves to muscles. This electrical activity can be recorded through an electromyography or EMG device, which will represent the electrical activity signal of the muscle in the form of a graph or signal of electrical activity over time. In each type of muscle movement, the resulting electrical activity will have different characteristics even though these movements occur in the same muscle [2].

One of the latest studies and research on EMG signal classification is the research by Ibrahimy et al in 2013 [3]. Ibrahimy in his research explains that a hand gesture in the upward, downward, right, and left movements has a specific EMG signal pattern or characteristic (with a single channel) that can be classified using an artificial neural network. Studies to make EMG signals into commands on a PC have been around since the early 1990s to 2000s, one of which was done by Barreto et al in 2000 [4]. In this research, Barreto et al used an "if" function on the EMG signal. This function will move the pointer if the recorded EMG signal on a specific electrode has exceeded the threshold set by the user beforehand. The other research on the two things mentioned above (using an artificial neural network as a translator of EMG signals into commands on a PC) has been conducted by Camilo Alexis Silva in 2014 [5]. The research used the same mechanism as used in Ibrahimy's research in 2013 [3], but with the output being a command on a PC (with the same activation mechanism as in Barreto et al's research (2000) [4]). Silva's research used an application inside a PC as a translator and command giver in the form of a pointer movement on a PC.

Along with the results of studies and research on EMG classification technology, the author is motivated to develop a device that can translate the classification results of movement directions in EMG signals into a command on the operating system of a PC. The device uses a human-computer interface mechanism based on electromyography signals. The device can be used to help or support computer operations for some users who do not have normal human capabilities in providing input commands on a PC, such as users who have movement disabilities in their upper extremities, especially fingers and hands. The main mechanism of the human-computer interface based on electromyography is the recording and translation of muscle signals in body movement parts. The muscle signal recorded by EMG can then be translated into a command to move a pointer on a PC operating system and can also provide specific commands such as inputting letters and numbers on a PC. In addition to requiring good EMG recording, this translation also requires an output signal translator module from EMG (in the form of a microcontroller) which will then provide simple on/off commands or movement directions on the computer interface that are appropriate and in line with what the user wants [3]. The muscles that will be recorded for their EMG signals as input to the human-computer interface in this study are the arm muscles

that have the capability to move the arm in four different directions (up, down, right, and left). These muscles are the brachioradialis and flexor carpi ulnaris [6]. The different bioelectric signal (EMG) activities in the four directions of muscle movement can be used as the basis for driving a computer interface such as a pointer, and the combination of the four directions of movement can be used as a specific command on the computer [5]. To obtain the EMG signal data from these muscles, an electromyography system is used in this study, consisting of a single-channel electrode, an Arduino Uno with an EMG shield, and a PC. To translate these EMG signals, an algorithm is needed to convert the EMG signals into digital signals that can be read by a PC as a command in the interfacing process. The module used in this study is an Arduino that is programmed using Microsoft Visual Studio 2013 (using Visual C# language). Arduino itself is a microcontroller that can receive input from the environment and can provide output to a device, such as an input to the PC interface in this study. By converting the EMG signal into digital data and then using a classification algorithm based on an artificial neural network in the embedded program embedded in the Arduino, a command that can be read by the operating system of a PC can be obtained.

The embedded program installed in the Arduino uses an artificial neural network algorithm in the interpretation of the direction of movement of the arm. By extracting the features contained in the recorded EMG signals and determining several outputs such as movement up, down, right, and left, a command can be created that can translate the characteristics of the EMG signal in each direction of arm movement into a command on the PC interface, such as the movement of the pointer. The artificial neural network parameters embedded in the Arduino program have previously been trained on a machine with sufficient capabilities, such as a PC, so the Arduino module only serves as a muscle signal recording and classification module.

The result of this study is a universal module that can translate the EMG signals of four-directional hand muscle movement (up, down, right, and left) as input to the PC interface. This final result can also be further developed into a robot arm, machine, and so on (for the development of artificial prostheses, and others) especially for neurorehabilitation purposes [7]–[10].

## 2. Materials and Methods

### 2.1 Device Block Diagram

This research can be broadly divided into two stages, each of which requires a different device configuration. The first stage involves preparing the device, such as acquiring EMG signal data. The device used during this stage is illustrated in the following diagram:

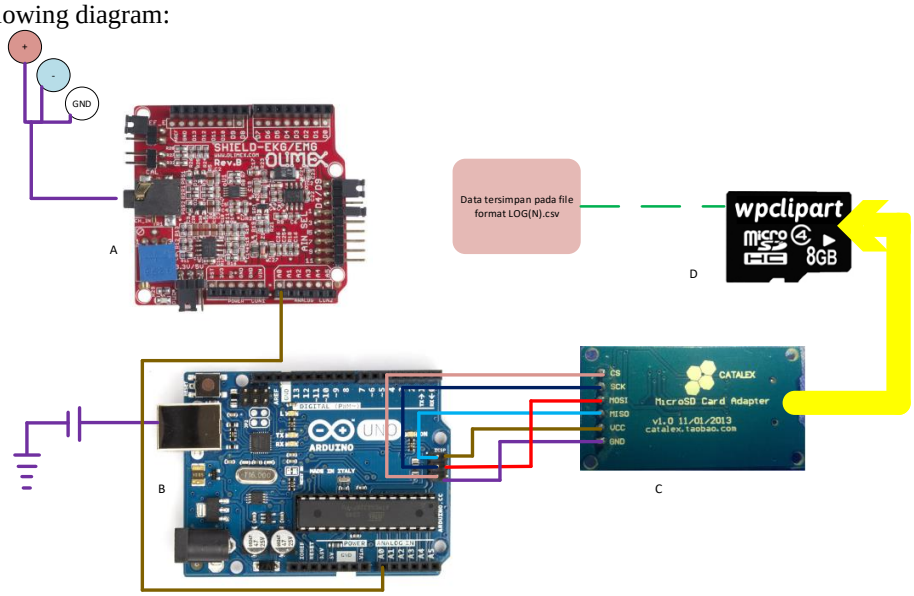


Figure 1. Schematic design of the device for the recording stage

In the second stage, the device functions as a translator for the EMG signal data and as an input device for the PC. The workflow of the device and the explanation of the usage stage can be found in Appendix 2. The block diagram configuration of the device is illustrated in the following diagram:

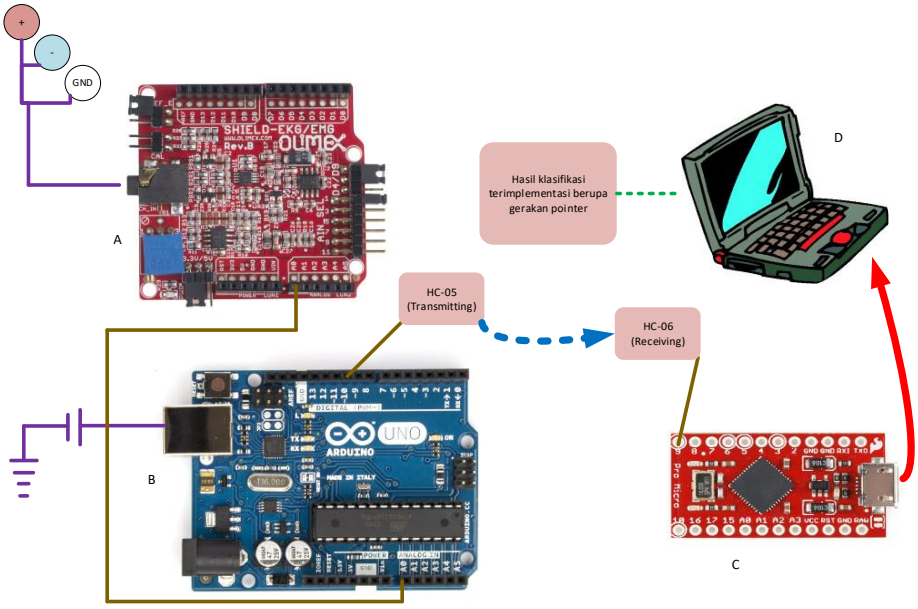


Figure 2 Schematic design of the device for the implementation stage.

2.2 Artificial Neural Network Architecture

In this study, the type of ANN training used is Backpropagation with the Levenberg-Marquardt, Scaled Conjugate Gradient and Normal Backpropagation as the optimization algorithms [3]. According to Ibrahimy et al. (2013), the best architecture that can be embedded in a microcontroller uses seven inputs in the form of extracted features from EMG signals, ten hidden layer nodes containing the tansig equation (hyperbolic tangent sigmoid transfer function), and an output layer containing the purelin equation (linear transfer function). The hyperbolic tangent sigmoid transfer function (tansig(n)) has the following formula and graph (mathwork.com, 2015):

$$tansig(n) = \left(\frac{2}{1+e^{-2\times n}}\right) - 1$$

On the other hand, the linear transfer function (purelin(n)) has the following formula and graph (mathwork.com, 2015):

$$purelin(n) = n$$

The ANN architecture used in this study is shown in Figure 3.

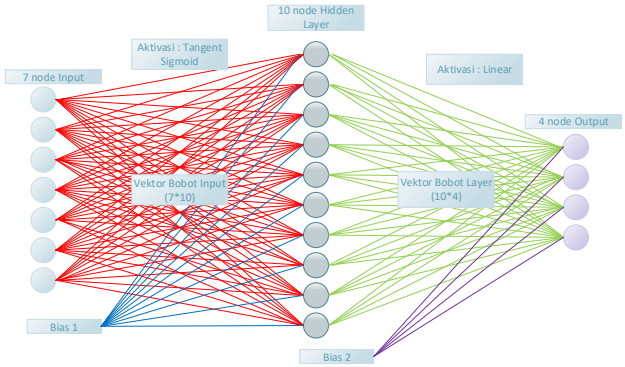


**Figure 3.** ANN with 7 inputs Ibrahimy *et. al.* (2013) [3]

Additionally, the ANN architecture used in Camilo Silva's 2014 [5] study on EMG classification, which inputs the raw EMG signal into the neural network without any feature extraction, is also used for comparison. The second architecture has a schematic as follows (with a reduced number of inputs adjusted to the capabilities of the Arduino Uno):

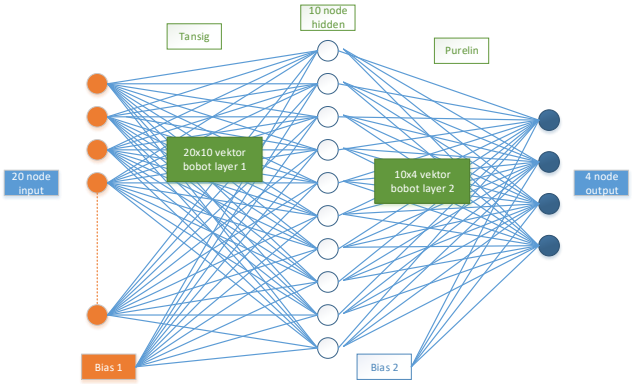


**Figure 4.** ANN with 20 inputs using raw EMG signals (Silva, 2014) [5].

Figure 4. shows the ANN architecture to be embedded in the Arduino Uno. The difference between the two ANN structures lies in the number and type of inputs used. In the first architecture, seven inputs are used, each of which is a feature that has been extracted from a signal (described in the next subsection). In contrast, the second architecture has 20 inputs, each of which represents the magnitude of the signal at 20 different positions (the 20 inputs form an EMG signal graph).

2.3 EMG Data Recording

In this study, data collection of EMG signals was conducted on one to six randomly selected individuals from among undergraduate students in the field of Biomedical Engineering (aged 18-22 years). Up to 10 data acquisitions were performed for each sample. The data acquisition points on the brachioradialis and flexor carpi ulnaris muscles, as well as the wrist as a reference point (as shown in Figure 5.), were further evaluated to determine if they had the same signal characteristics for each movement.

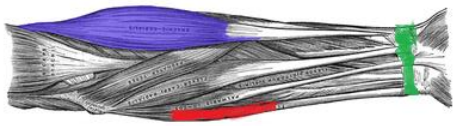

**Figure 5.** The placement points of the EMG electrodes are indicated in red for the first acquisition point (flexor carpum ulnaris), purple for the second acquisition point (brachioradialis), and green for the reference point [3], [11].

Subsequently, with the activation of the data recording function in the program, volunteer subjects were instructed to move their hands according to the given commands. The recorded data was then saved on a PC in the form of digital signals (consisting of data Xn = signal amplitude at n and N = length of the recorded digital signal), which were then subjected to feature extraction. The recorded EMG signals were generated by performing hand gestures in several directions, namely: up, down, right, and left. The gesture shapes that were classified are: radial deviation was used for upward movement, ulnar deviation for downward movement, supination for rightward movement, and pronation for leftward movement [12].

2.4 EMG Feature Extraction

According to the previously planned ANN architecture, there are seven features extracted from the recorded EMG signal. The extracted features along with their respective equations (Mean Average Values, Root Mean Squares, Variance, Standard Deviations, Wavelength, Zero Crossing, Slope Sign Change) are as follows [3], [13]:

$$MAV = \frac{1}{N}\sum_{n=1}^{N} |x_{n^2}| \quad RMS = \sqrt{\frac{1}{N}\sum_{n=1}^{N} x_n^2} \quad VAR = \frac{1}{N}\sum_{n=1}^{N}(x_n - \bar{x})^2 \quad SD = \sqrt{\frac{1}{N-1}\sum_{n=1}^{N}(x_n - \bar{x})^2}$$

$$WL = \frac{1}{N}\sum_{n=1}^{N-1} |x_{n+1} - x_n| \quad ZC = \sum_{n=1}^{N-1}[sgn[(x_n \times x_{n+1}) \cap |x_n - x_{n+1}|] \geq threshold]$$

$$SSC = \sum_{n=1}^{N-1}[f[(x_n \times x_{n+1}) \times (x_n - x_{n+1})]]$$

2.5 ANN Training

The input signals are fed into the ANNs [14], and 50 sets of feature data for each type of EMG signal (up, down, right, and left movements) are processed through the network architecture equations to produce an output matrix that approximates the values in the Table 1.

**Table 1** ANN Output Matrix

| | Up | Down | Right | Left |
|---|---|---|---|---|
| Matrix Target | 1 | 0 | 0 | 0 |
| | 0 | 1 | 0 | 0 |
| | 0 | 0 | 1 | 0 |
| | 0 | 0 | 0 | 1 |

To improve the efficiency of the training process and avoid overfitting, the data sets are divided into 90% for training and 10% for validation. Two stop criteria are given: the training process will stop when the error percentage is less than or equal to 0.001 (using the MSE parameter) or has reached 1000 epochs in one cycle, and when the error resulting from the validation data input exceeds 0.15 (using the MSE parameter) [15].

2.6 Implementation and Error Rate Calculation

At this stage, the results of the ANN embedded in the Arduino Uno will be used as input matrices for up, down, right, and left movements on the Sparkfun Pro Micro. The Sparkfun Pro Micro will then translate these matrices into pointer movement commands based on the direction of the matrix. Error calculation in this mechanism is divided into two parts: offline processing, by feeding previously recorded data to the ANN architecture, and online processing, by recording EMG signals and classifying the results into an output on the PC. Online testing used pointer movement on the PC to evaluate the classification results. To determine the amount of error in the online test, a large pointer movement distance calculator application was used. The test results were then analyzed for the deviation in each movement by examining the coordinates before and after the movement, with the control value being the initial coordinates plus 100 pixels (according to the direction of hand gesture). Additionally, the amount of time taken to reach 100 pixels was also shown to evaluate the implementation performance.

## 3. Results and Discussion

3.1 ANN Model Selection

Following the research methodology, a total of 210 sets of EMG signals were recorded using a recording circuit with a sampling frequency of approximately 100 Hz. The recorded signals were divided into four classes corresponding to upward, downward, rightward, and leftward hand movements. The recorded data were not calibrated into millivolt units but were kept in the range of the DAC on the Arduino Uno (0-1023), to facilitate feature extraction and classification by the artificial neural network. This study focuses on implementation rather than monitoring the EMG signal values. With the extracted features mentioned above, both types of data sets were used as input for each neural network structure to perform the training mechanism to obtain neural networks with weight values that can classify in real-time during implementation. To be easily readable by the neural network, data from all extracted features were normalized with a range of -1 to 1. With the results of feature extraction as described above, both types of datasets are used as inputs to each neural network to undergo training mechanisms to obtain neural networks with weight values that can classify in real-time during implementation. As per the research methodology described, there are two neural network structures, the first of which has seven inputs, ten hidden nodes, and four outputs, while the second has twenty inputs, ten hidden nodes, and four outputs. The output results in a matrix with a range of values from -1 to 1. The training mechanism uses a Signal Classification application. The training algorithm used is Levenberg Marquardt with training parameters such as rate set to default values. To compare the performance of the training algorithms, the use of normal backpropagation and scaled conjugate gradient algorithms is also applied. By comparing the performance results, the optimal variable for use as the training parameter for the neural network can be determined. Table 2. presents the training results in terms of error and number of epochs, along with variables such as number of inputs and type of training (each type of training repeated up to three times).

**Table 2.** The Training Performance with Several Parameters

| No | N Input | Training Method | Validation Error | Best Epoch | Training Time |
|---|---|---|---|---|---|
| 1 | 7 | *Scaled Conjugate Gradient* | 0.0785420130510243 | 97 | 215ms |
| 2 | 7 | *Levenberg Marquardt* | 0.046550741454159 | 30 | 2673ms |
| 3 | 7 | *Normal Backpro* | Validation not satisfied, stopped at E= 4.99782709062222E+108 | - | - |
| 4 | 20 | *Scaled Conjugate Gradient* | 0.024576991655724 | 79 | 244ms |
| 5 | 20 | *Levenberg Marquardt* | 0.0517344237565072 | 10 | 3704ms |
| 6 | 20 | *Normal Backpro* | Validation not satisfied, stopped at E= 1.69617971751728E+168 | - | - |

Based on the training results shown in the above table, it is evident that the use of normal backpropagation method (backpropagation method with weight updates using error gradient) requires more than 1000 epochs to achieve good results. The best performance is achieved with the Levenberg-Marquardt method, which requires relatively fewer iterations to reach a training data MSE margin below 0.001 or the best validation data MSE. The table also shows that the second neural network structure provides excellent performance, especially with the Levenberg Marquardt training method, which requires at least less than 50 iterations to reach the minimum MSE. The parameters resulting from the previous training process are then saved for each type and repetition of the training method. The saved parameters from the previous subsection are then tested on several EMG signals outside of the training and validation datasets. This offline test is performed using the Offline Test application described in methodology. The test results are summarized in the following Table 3. (the results with the Normal Backpropagation method are not used because they do not meet the minimum validation error requirements).

**Tabel 3.** Offline Implementation Test

| No | N Input | Training Method | MSE | Classification Rate |
|---|---|---|---|---|
| 1 | 7 | *Scaled Conjugate Gradient* | 0.188383833127176 | 68.75% |
| 2 | 7 | *Levenberg Marquardt* | 0.311765581662042 | 56.25% |
| 3 | 20 | *Scaled Conjugate Gradient* | 0.246904404051887 | 81.25% |
| 4 | 20 | *Levenberg Marquardt* | 0.288058281063636 | 75% |

The table above reveals two profiles or sets of training parameters that achieved the best classification percentage (above 70%). The first profile is Scaled Conjugate Gradient (20 inputs) with a percentage of 81.25%, while the second is Levenberg Marquardt (20 inputs) with a percentage of 75%. This finding also suggests that the second neural network structure (20 inputs, 10 hidden, and 4 outputs) [5] resulted in better classification performance. Based on these offline test results, both profiles of saved parameters were chosen for further real-time testing.

3.2 Online Implementation Test Results

The online test results can be seen in Table 4. The data were taken by calculating the deviations of pointer movements after the gestures are conducted.

Tabel 4 Real-Time Implementation Test Results

| No | Gestures | ANN Profiles | X Coordinate Deviation Rates | Y Coordinate Deviation Rates |
|---|---|---|---|---|
| 1 | Atas | *Scaled Conjugate Gradient (20 input)* | 4.2 | 6.4 |

| | | | | |
|---|---|---|---|---|
| 2 | Bawah | *Scaled Conjugate Gradient (20 input)* | 3.5 | 4.3 |
| 3 | Kanan | *Scaled Conjugate Gradient (20 input)* | 15.9 | 6.9 |
| 4 | Kiri | *Scaled Conjugate Gradient (20 input)* | 11.5 | 8.8 |
| 5 | Atas | *Levenberg Marquardt (20 input)* | 3.3 | 3.6 |
| 6 | Bawah | *Levenberg Marquardt (20 input)* | 6.5 | 6.7 |
| 7 | Kanan | *Levenberg Marquardt (20 input)* | 11 | 7.3 |
| 8 | Kiri | *Levenberg Marquardt (20 input)* | 10 | 6.1 |

The calculation above reveals that the classification rate for profile 1 is 91.7625%, and for profile 2 is 93.7375%. For each gesture: with success rates of 96.55% for upward direction, 93.4% for downward direction, 90.85% for rightward direction, and 91.95% for leftward direction.

## 4. Conclusion

The algorithm used to activate the electromyography-based computer operation function is a neural network classification algorithm with twenty inputs, ten nodes in the hidden layer, and four outputs representing basic movement directions (up, down, right, and left). The neural network was previously trained using the Levenberg-Marquardt and Scaled Conjugate Gradient methods. During implementation, the algorithms achieve classification success rates of 93.7375%, with success rates of 96.55% for upward movement, 93.4% for downward movement, 90.85% for rightward movement, and 91.95% for leftward movement.

## References

[1] E. N. Marieb and K. Hoehn, *Human Anatomy & Physiology*. San Francisco: Benjamin Cummings, 2010.
[2] G. Kamen, "Electromyographic Kinesiology," in *Research Methods in Biomechanics*, D. G. E. Robertson and others, Eds., Champaign, IL: Human Kinetics Publ., 2004, pp. 227–258.
[3] M. I. Ibrahimy, M. R. Ahsan, and O. O. Khalifa, "Design and Performance Analysis of Artificial Neural Network for Hand Motion Detection from EMG Signals," *World Appl. Sci. J.*, vol. 23, no. 10, pp. 1322–1329, 2013.
[4] A. Barreto, S. D. Scargle, and M. Adjouadi, "A practical EMG-based human-computer interface for users with motor disabilities," *J. Rehabil. Res. Dev.*, vol. 37, pp. 53–63, Nov. 1999.
[5] S. CA, *Design of Bioinstrumentation for Monitoring and Classification of Biosignal, ECG and EMG in Real Time*. 2014.
[6] J. A. Raszewski, A. C. Black, and M. Varacallo, "Anatomy, Shoulder and Upper Limb, Hand Compartments," *StatPearls*, Sep. 2022, Accessed: Dec. 09, 2022. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK532942/
[7] M. Kuzuya, "Process of Physical Disability Among Older Adults-Contribution of Frailty in The Super-Aged Society," *Med. Sci. J. Nagoya Unversity Grad. Sch. Med.*, vol. 74, pp. 1–15, 2012.
[8] V. Kumar, *Robbins and Cotran Pathologic Basis of Disease*. Philadelphia, PA: Saunders/Elsevier, 2010.
[9] G. A. Donnan, M. Fisher, M. Macleod, and S. M. Davis, "Stroke," *Lancet*, vol. 371, no. 9624, pp. 1612–1623, 2008.
[10] F. I. Serdana, "Controlling 3D Model of Human Hand Exploiting Synergistic Activation of The Upper Limb Muscles," *IES 2022 - 2022 Int. Electron. Symp. Energy Dev. Clim. Change Solut. Clean Energy Transit. Proceeding*, pp. 142–149, 2022, doi: 10.1109/IES55876.2022.9888488.
[11] B. S. Bowden and J. M. Burke, *An Illustrated Atlas of Skeletal Muscle*. New Jersey: McGraw-Hill, 2002.
[12] E. J. Weiss and M. Flanders, "Muscular and postural synergies of the human hand," *J Neurophysiol*, vol. 92, no. 1, pp. 523–535, 2004.
[13] M. R. Ahsan, M. I. Ibrahimy, and O. O. Khalifa, "Neural Network Classifier for Hand Motion Detection from EMG Signal," *IFMBE Proc.*, vol. 35, pp. 1045–1048, 2011.
[14] R. DE, H. GE, and W. RJ, *Learning representations by back-propagating errors*. Nature Journal, 1986.
[15] M. Hioki and H. Kawasaki, "Estimation of Finger Joint Angles from sEMG Using a Neural Network Including Time Delay Factor and Recurrent Structure," *ISRN Rehabil*, vol. 2012, pp. 1–13, 2012.