

Hand Gestures Robotic Control Based on Computer Vision

Marthed Wameed¹, Dr. Ahmed M. Alkamachi²

Submitted: 22/11/2022

Revised: 23/01/2023

Accepted: 19/02/2023

Abstract: Hand gestures are considered one of the most important and simple ways of communicating between people and robots, especially for humans who suffer from speech and hearing difficulties (the deaf and dumb). Sign language (hand gestures) is used to communicate with them. In this research, the proposed system consists of two parts: The first part is the detection and classification of hand gestures in real time using computer vision technology; this is done by machine learning, specifically the MediaPipe algorithm. The MediaPipe algorithm consists of three sections: the first is the detection of the palm of the hand; the second is identifying 21 points 3D on the palm; and the third is the classification of hand gestures, which is done by comparison between the dimensions of those points. The second part, which depends on the first part, stipulates, after detecting and classifying the hand gestures, the system controls the robot through hand gestures, as each hand gesture has a specific movement that the robot performs. The experimental results showed through the effect of environmental elements such as light intensity, distance, and tilt angle (between hand gesture and camera) that the proposed system can perform well in controlling the movement of the robot through hand gestures.

Keywords: Computer vision, Machine learning, MediaPipe, Hand landmarks.

1. Introduction

In recent years, the detection and classification of gestures have attracted great interest, especially in the field of human-computer interaction (HCI). Human dependence on computers has increased in daily activities, so there is a growing need to develop methods of communication between computers and humans that do not rely on the traditional method. Gestures are the only way to communicate with people who are deaf and dumb (special needs), so recognizing the gestures makes it easy to communicate with them or communicate with the machine. There are several ways to detect and classify hand gestures in this work it relied on computer vision specifically the machine learning algorithm (MediaPipe), is utilized. The techniques of turning images into digital data and altering their nature so that they are easier for machines to grasp while simultaneously enhancing the visual information and human perception of it are known as computer vision [1]. Computer vision is the only technique that can extract information from incoming images [2]. Figure 1 shows the methods used and widely used for gesture application. Detection of hand gestures based on computer vision has several advantages compared to gloves that contain sensors. Different kinds of sensors were used in the gloves to track

hand movement and position by finding the exact coordinates of where the hand and palms were [3]. A number of different sensors, such as the curvature sensor [4], the displacement sensor [5], the fibre optic transducer [6], the flexible sensors [7], and the gyroscopes [8]. These sensors rely on a variety of physical principles.

The electrical connections in gloves and the cost of maintaining sensors make it hard for people to use them. Last, a lot of people will use gloves, which makes it easier for skin diseases to spread from one person to another [9] because of this, the computer vision method was used in this work. The purpose of this manuscript is to utilize computer vision to detect and classify hand gestures from a webcam. This helps people with special needs to communicate with people or communicate with the machine without any problem. There are other methods that depend on computer vision specifically deep learning (DL) in detecting and classify hand gestures [10-12]. These methods are considered complex because they require a very large dataset and thus consume more time in data processing.

¹ Department of Mechatronics Engineering, Al-khwarizmi College of Engineering, University of Baghdad
ORCID ID: 0000-0001-9822-6179

² Department of Mechatronics Engineering, Al-khwarizmi College of Engineering, University of Baghdad
ORCID ID: 0009-0008-7784-0012

* Corresponding Author Email:

Marthad.Wameed1202a@kecbu.uobaghdad.edu.iq

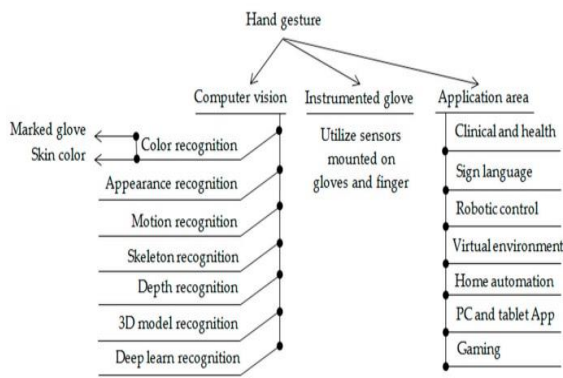


Fig 1. Methods for detecting hand gestures and their applications [7].

2. Related Works

In this section, a summary of previous studies related to the topic of our work using computer vision is presented follows:

Yuan Chung, et al. [13], the aim of this research is to use the webcam to detect and classify hand gestures to control home appliances (smart home) through several stages: first, skin color detection and morphology are used to remove unwanted information from the image. Second: background subtraction is used to detect the region of interest (ROI) and finally the image size is changed to 100 * 120 and then it is inserted into the convolutional neural network (CNN). In this work two types of convolutional neural network (CNN) are used VGGNet and AlexNet. **Waskito et al. [14]**, through this research, hand gestures are classified and detected using deep learning, specifically using a convolutional neural network (CNN). After detecting the hand gestures, the system controls the robot's movement wirelessly through those gestures. They use six hand gestures to control the robot's movement. Their dataset consisted of 1000 images, the size of these images is 330 * 330. **Huu et al. [15]**, used image processing and an artificial neural network (ANN) to detect and classify dynamic hand gestures. This is done through three steps: hand detection, hand tracking and hand recognition, and then controls the smart home through the camera connected to the computer that takes pictures of the dynamic hand gestures. **Zhang, et al. [16]**, deals with augmented reality (AR) and virtual reality (VR) and how to control them through hand gestures in the real time based on machine learning, specifically using the MediaPipe algorithm, and this is done through three stages: First stage detecting a palm by defining the area of the palm in the image and surrounding it with a rectangle. Second stage the hand Landmarks, in which 21 3D (X, Y, Z) points are identified on the hand. Third stage the classification hand gestures by comparing the dimensions of those points. Worth mentioning first stage (detecting a palm) runs only on first frame or when hand is missing

.Taban, et al. [17] , Depending on the computer vision, specifically the use of the Viola-Jones algorithm to detect and classify hand gestures, and through these gestures, the lights are turned on and off. The lighting response time to the gesture commands was 0.43 seconds, and the dataset consisted of eight gestures containing 12000 images. In this work, for more efficient and precise computer training, a novel concept proposes using skin detection prior to computer training to automatically establish the size and position of all positive pictures. **Boruah, et al. [18]**, the research deals with modern methods of communication between the teacher and the student through controlling mouse cursor (virtual-mouse) without the need for electronic equipment, and it is done through hand gestures. Where 6 hand gestures are used, and these gestures are detected and classified by computer vision, specifically using the MediaPipe algorithm, and each hand gesture has a specific command that is executed by the mouse cursor without using electronic equipment (computer pointer).

3. System Design

As a result of the system's implementation, a tracked robot's movement can be guided in real time by means of hand gestures. In this study, we will use the MediaPipe algorithm to detect and classify hand gestures based on 21 3D points (landmarks) and then send commands to control the robot's movement, as each hand gesture has a unique command.

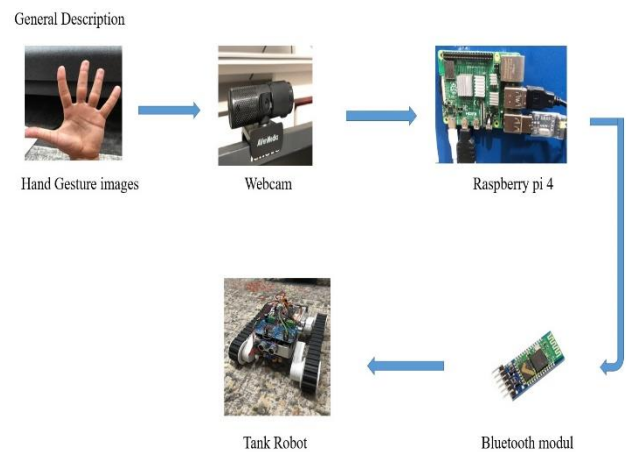


Fig 2. Explain the components of the system

According to Figure (2), it shows an overview of the working principle of the system and its parts. The images will be taken by a webcam camera (AverMedia PW313) and then processed on a Raspberry pi 4 processor. On the Raspberry pi 4, image feature extraction as well as picture classification will be carried out. The MediaPipe algorithm will be used to detect and classify hand gestures image according to the dataset that has been provided by using Raspberry Pi 4 Microcontroller. The Raspberry pi 4 will be connected to a Bluetooth transmitter, which will then

broadcast the results of the classification in a wireless manner. The results (output) are received by the Bluetooth module that is installed within the tracked robot, As a result, the tracked robot moves through the commands received. Figure (3) shows the pattern and the path that the tracked robot will follow as it moves.






| The Form of Hand Gestures | | | | |
|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 |
|  |  |  |  |  |
| The Commands Carried Out in the Robot | | | | |
| Stop (Robot is not moving) | Move forward | Move backward | Left (Rotating anti clockwise) | Right (Rotating clockwise) |

Fig. 3 The form of hand gestures and commands carried out

As it can be seen from Figure (3), there are five types of hand gestures used in this work to control tracked robot.

3.1. Hardware Setup

The hardware Setup consists of two parts: the tracked robot and the base station, as shown in Figure (4).

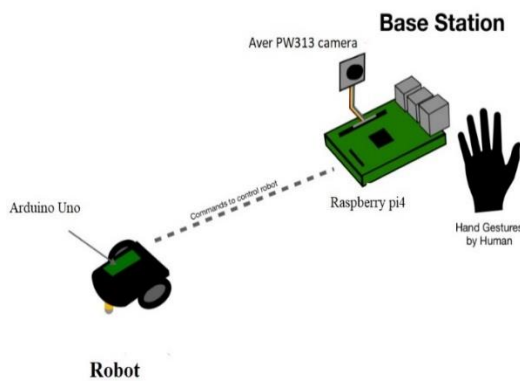


Fig 4. Diagram of system parts.

3.1.1 Tracked Robot Components

1. Microcontroller type Arduino Mega.
2. HC-SR04 Ultrasonic Range Sensor.
3. Bluetooth HC-05.
4. Battery Pack and switch
5. DC-DC Step down 5A XL4015
6. Motor Driver BTS7960.
7. Two DC motors

The components of the tracked robot can be illustrated by the Figure (5).

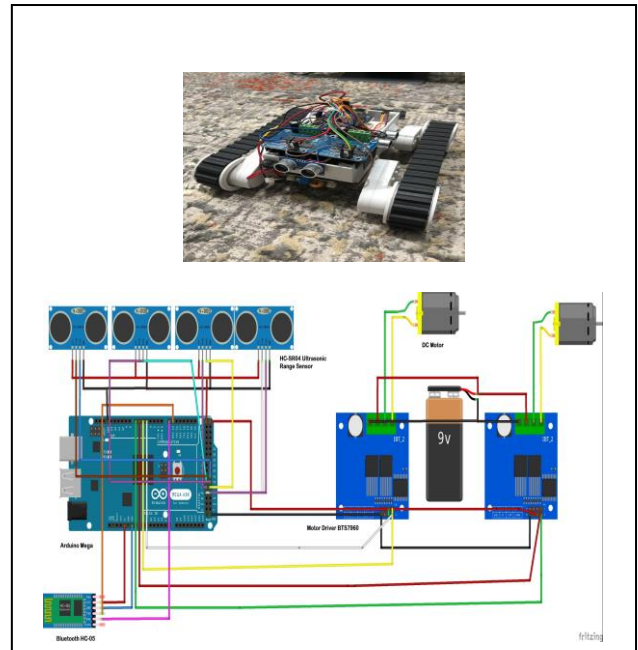


Fig. 5. The assembled Tank robot.

3.1.2 Base Station Components

1. An 8MP 1080 p30 resolution Aver PW313 camera powered by the Microcontroller.
2. Microcontroller type Raspberry pi4 8GB.
3. Bluetooth HC-05.
4. LCD screen

The components of the base station can be illustrated by the Figure (6)

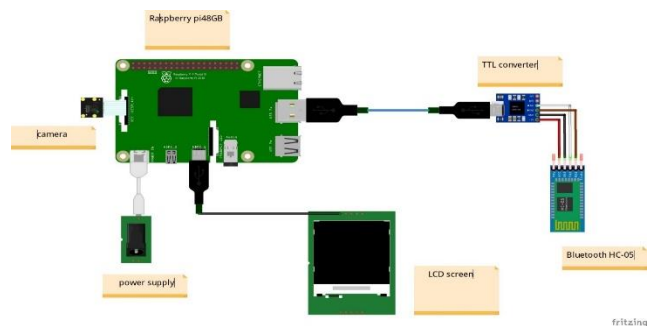


Fig. 6 Base station parts description.

3.2. Software Configuration

The software was separated into two categories: base station software and robot software.

3.2.1 Base station software

The software for the base station focused on recognizing different kinds of gestures. To begin achieving this goal, the following steps represent the software configuration into the base station (Raspberry pi 4):

1. Download the Raspberry Pi imager application from the Raspberry pi Website [19].
2. Install the operating system (Raspberry Pi OS 32 bit) inside the SD (Secure Digital) card from the Raspberry Pi imager application
3. Input the SD card into the Raspberry Pi, and now the Raspberry Pi is ready to work
4. The last step is to install Python packages like torch2trt, JetCam as well as download the OpenCV and the MediaPipe library inside the Raspberry pi

3.2.2 Robot software

The microcontroller used in the tracked robot is an Arduino Mega type to control the movement of the DC motors through the commands received from base station. The Arduino C programming language and the Arduino IDE software are used in the design of Arduino's software.

4. Methodology

In the process of detecting the hand, the palm hand is first detected in the image or video clip. This is done by machine learning using the MediaPipe algorithm with computer vision or image processing.

4.1 MediaPipe Framework

In July of 2019, Google published MediaPipe as an open-source tool. This method detects the palm with the use of Single shot detection (SSD), an object identification technique(detection the palm), and the hand landmark model is used to extrapolate the information necessary to recognize the fingers .The MediaPipe is an algorithm that detects individual hands inside a video stream. The algorithm is developed using machine learning techniques .Real-time hand tracking is possible by using MediaPipe for everyone with a webcam. By using machine learning, technique in this study extrapolates 21 points 3D from a single picture. Real-time tracking of several hands is made possible by a three-stage approach consisting of a palm detection model, a hand landmark model and gesture recognizer model as shown in Figure(7) [16][20][21][22].The three models, as well as the training datasets, are detailed in more depth below.

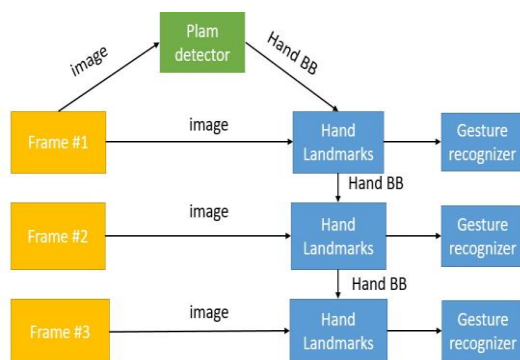


Fig. 7. MediaPipe algorithm parts.

4.1.2 Datasets

Three distinct data sets are selected for use in the training of the MediaPipe models. The description of the datasets that are used in the training of the model is listed below.

- **The wild dataset:** This dataset contains 6,000 pictures that vary greatly in regard to factors like location, lighting, and the way people's hands look. One drawback of the dataset is that it doesn't capture more complex forms of hand articulation.
- **The house collected dataset:** There are around 10,000 images in this collection, each showing a different hand movement from every possible angle. The disadvantage of this dataset is that it was collected from just 30 persons, and there was very little variation in the backgrounds of those people.
- **Synthetic dataset:** This dataset takes a high-resolution representation of a synthetic hand model and maps it to its 3D coordinates against a variety of backgrounds. Five skin tones are included in the hand model's texturing. One hundred thousand still images were captured from the change video streams between hand positions. Three cameras were used to capture each pose in a randomly generated high dynamic range lighting scenario.

4.1.3 Palm Detector Model

Compared to other tasks, detecting hands is more difficult since it must be able to distinguish between different hand sizes and recognize closed portions of hands. It is possible to use the contrast and forms of a person's nose and lips as distinguishing traits in a face detection activity. In contrast, comparable traits are absent in the hand detection task, significantly increasing its difficulty. Thus, the palm detection is trained since it is simpler to estimate bounding boxes around stiff objects like the palm and the fist [16].

4.1.4 Hand Landmark

After revealing the Palm Detector model, the Landmark stage comes. The 21 three-dimensional points are identified on the hand, as shown in Figure (7). Where the X-axis represents the width of the image, the Y-axis represents the height of the image, and the Z-axis represents the depth of the image (the distance between the hand and camera).Comparing between the dimensions of 21 points 3D, the hand gestures are classified.

4.1.5 Gesture Recognizer Model

Figure (8) shows 21 points 3D on each part of the palm. To classify hand gestures, the value of coordinates of the points (4, 8, 12, 16, 20) are compared. These points represent the fingertips with the value of coordinates of the points (3, 5, 9, 13, 17) that represent the middle points relative to the x and y axes.

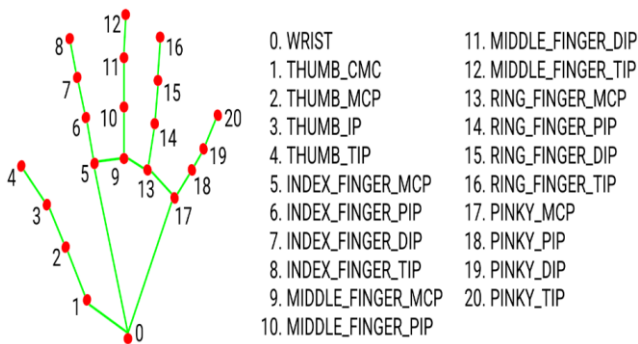


Fig. 8. Hand Landmark [20].

- **The state of the thumb.**

The thumb is open if the value of coordinate of the Point (4), (which represents fingertip) relative to the X-axis, is greater than the value of coordinate of the Point (3), (which represents the middle point)relative to the X-axis, and vice versa.

- **Second, the state of the all fingers except for the thumb.**

The fingers (index, middle, ring, and pinky) are opens if the values of coordinates of the points (8, 12, 16, 20), (which represent the fingertip) relative to the Y axis are greater than the values of coordinates of the points (5, 9, 13, 17), (which represent the middle point) relative to the Y axis, and vice versa.

Figure (9) illustrates the graph that we created in MediaPipe in order to monitor hand movements. The graph is segmented into two parts: one to detect hands and another to calculate landmarks [16]. One of the main improvements provided by MediaPipe is the palm detector, which only executes when required (which isn't very frequently) and therefore dramatically lowers computing costs. Instead of using the palm detector on every frame, we accomplish this by determining the hand location in the current video frames from the computed hand landmarks in the previous frame. The main contribution in this manuscript is to improve the MediaPipe algorithm in a way that makes it the fastest response in detecting hand gestures in real time by making the hand detection only run on the first frame or when the hand is missing in the image. Which significantly reduces response time and makes the algorithm more accurate at detecting hand gestures.

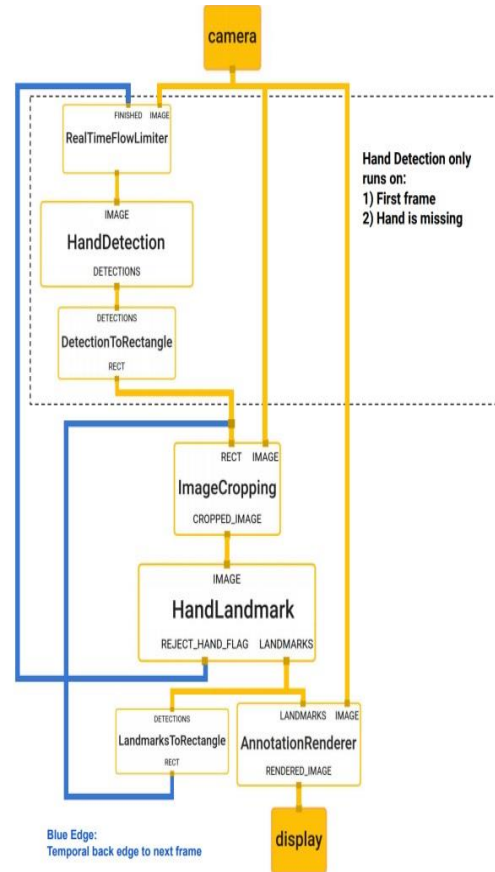


Fig. 9. Stricture of MediaPipe algorithm [16].

4.1.6 Configuration Settings of MediaPipe Algorithm

There are a variety of configuration settings that may be applied. The configuration are explained below:

Static_Image_Mode: a flag that can be used to determine which mode will be utilized while reading the image. There are two modes:

- Default mode, depending on the data set that are previously collected for testing the proposed model.
- Undefault mode depending on the data set from real time for testing the proposed model.

In this study, the proposed model are tested in real time using Undefault modes

Max_Num Hands: Depending on the number of hands being examined, a value for this integer variable must be determined.

For this work, the default value of 2 is used.

Model_Complexity: This flag indicates the accuracy and response time. If the value is 1, this indicates that the system works with high accuracy, but with a longer response time.

If it is 0, it indicates the opposite. In this work, the value was set to the default value, which is 1.

Min_Detection_Confidence: This is the level of confidence that the hand detector uses to set the output threshold.

If the confidence is less than this provided threshold value, the detectors is considered to have failed, and also no output is given.

The number must be between 0.0 and 1.0. In this work, the default value of 0.5 is utilized.

Mim_Tracking_Confidence: The static mode does not make use of this flag. When performing real-time hand tracking, this value should be set between zero and one. This is done to ensure that the hand is successfully monitored. This uses the default of 0.5. During live tracking, if the confidence is too low, it will move on to the next image without doing the detection. Hand detection is performed on each and every image in static mode [20].

5. Experimental Results

5.1 Test the Proposed Model Control the Robot in Real Time

Depending on the changing intensity of illumination, the accuracy of the system is calculated this is the first testing for the proposed system. A specific application is used to measure the intensity of lighting based on the phone's camera. In this research, lighting with a variable intensity that is 50 lux, 100 lux, and 150 lux, has been used. Additionally, environmental settings such as the distance between hand gestures and the camera being 1 m and the tilt angle between hand gestures and the camera being 0° have been specified. The experiment is repeated 20 times for each hand gesture and light intensity.

Table 1. Light intensity variation based accuracy

| Hand Gesture | Accuracy with Intensity Light (%) | | |
|--------------|-----------------------------------|---------|---------|
| | 50 LUX | 100 LUX | 150 LUX |
| Stop | 100 | 100 | 100 |
| Forward | 100 | 100 | 100 |
| Backward | 100 | 100 | 100 |
| Left | 75 | 100 | 65 |
| Right | 65 | 100 | 40 |

According to Table (1), when the illumination level is set to 100 lux, the accuracy for all hand gestures reaches 100%.

Depending on the changing the distance between camera and hand gestures, the accuracy of the system is calculated.

This is the second test for the proposed system. The measurement distance is the distance between the hand gestures and the camera. The distances chosen are 1 m, 2 m, and 4 m. additionally, environmental settings such as the intensity of light being 100 lux and the tilt angle between hand gestures and the camera being 0° have been specified. Experiment is repeated 20 times for each hand gesture and specified distance.

Table 2. Variations in distance-based accuracy

| Hand Gesture | Accuracy with Distance (%) | | |
|--------------|----------------------------|-----|-----|
| | 1 m | 2 m | 4m |
| Stop | 100 | 100 | 100 |
| Forward | 100 | 100 | 100 |
| Backward | 100 | 100 | 100 |
| Left | 100 | 80 | 70 |
| Right | 100 | 70 | 70 |

Table (2) shows that at 1 m, the accuracy reaches 100% for all hand gestures. This means that at this distance, all hand gestures can be correctly classified.

Depending on the changing tilt angle between camera and hand gestures as shown in Figure (10).The accuracy of the system is calculated. This is the second testing for the proposed system.

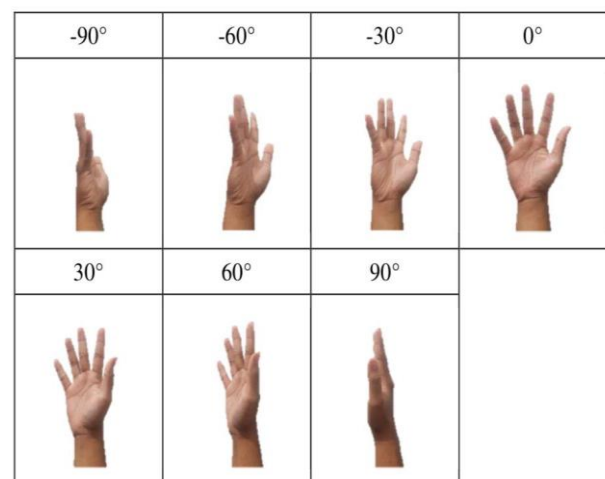


Fig. 10. The tilt angles between the hand and the camera.

The tilt angle used between the hand gesture and the camera is 0°, 30°, 60° and 90° in this test. Additionally, environmental settings such as the distance between hand gestures and the camera being 1 m and intensity of light being 100 lux. Each hand gesture and tilt angle is tested 20 times.

Table 3. Tilt-angle testing accuracy

| Hand Gesture | Accuracy with Distance (%) | | | |
|--------------|----------------------------|-----|-----|-----|
| | 0° | 30° | 60° | 90° |
| Stop | 100 | 100 | 100 | 60 |
| Forward | 100 | 100 | 100 | 30 |
| Backward | 100 | 100 | 100 | 20 |
| Left | 100 | 100 | 55 | 15 |
| Right | 100 | 100 | 40 | 12 |

Table (3) shows high accuracy values, with 100% for each hand gesture when the slope angle is 30° degrees or 0° degrees.

5.2 Actuators Power Consumption Test

Two electric motors (6v, DC) are used to move the tracked robot, while a 2,200-mAh, 14.8-volt Li Ion battery is used to power the motors. Various hand gestures have different PWM values.

Table 4. Actuators Power Consumption Test for each pulse width modulator (PWM)

| Hand Gesture | Left Motor | | Right Motor | |
|--------------|------------|-------------------|-------------|-------------------|
| | PWM | Consumed Power(w) | PWM | Consumed Power(w) |
| Stop | 0 | 0 | 0 | 0 |
| Forward | 38 | 0.808 | 38 | 0.8602 |
| Backward | 38 | 0.8064 | 38 | 0.808 |
| Left | 29 | 0 | 29 | 0.6768 |
| Right | 29 | 0.633 | 29 | 0 |

Using current sensor (INA219) to measure consumed power in the above table, a slight difference in the value of consumed power for the same value of PWM between the right and left motors is noticed. This difference is due to the distribution of loads in the robot.

5.3 Time Response

Time testing checks how long it takes from when the input is given until the output is run. Hand gestures are used as input, and the robot's motors move as a result. The response time is calculated from the time the input is given, and then the Raspberry Pi 4 processes the data and sends the data to the robot motors via Bluetooth. After that, each hand gesture undergoes a test of response time. Each hand gestures is tested 3 times.

Table 4. The Robotic Response time test

| Hand Gesture | Average Reaction Time (s) |
|---------------------------------|---------------------------|
| Stop | 0.091 |
| Forward | 0.121 |
| Backward | 0.126 |
| Left | 0.162 |
| Right | 0.165 |
| Average Total Reaction Time (s) | 0.133 |

Table (4) shows that the average time it takes for hand gesture input to trigger motor action on the robot is 0.133 seconds.

6. Discussion

The accuracy of some commands decreases with changing environmental factors in real time. That is due to the system's algorithm not being able to correctly classify hand gestures. The most important reason for the system's inability to classify hand gestures correctly is that the fingers are close together and some fingers are not fully closed or opened. . That may present a delay in the response of the robot's movement to some commands. Thus, changing some irregular hand gestures to more regular gestures to increase accuracy and reduce the robot's response time is recommended.

7. Conclusion

This study proposed a model to control the movement of the robot through hand gestures using computer vision, specifically relying on the MP algorithm. The MP algorithm classifies hand gestures by comparing the coordinates of 21 three-dimensional points specified on the hand. The data sets used in this study for testing Proposed Model consisted of 16000 images of various hand gestures captured under various environmental conditions. Through this model, the robot's movement was very well controlled, but there were some gestures that were not accurately classified in real time. The reason is due to the fact that some cases, the fingers are close together, and some fingers are not fully closed or open. That leads to the inability of the algorithm used to classify hand gestures correctly, and thus the classification accuracy decreases. That gives impossibility for the system to determine whether the finger is closed or opened. Hence, the system depends on the classification of hand gestures based on the comparison between the coordinates of the 3D points specified on the hand. To increase the accuracy of the classification of hand

gestures, suggestion of replacing these with others in which the fingers are more clearly defined. That leads to more accurate classification and thus reduces the response time of the robot to commands.

References

- [1] F. Hardan and A. R. J. Almusawi, "Developing an Automated Vision System for Maintaing Social Distancing to Cure the Pandemic," *Al-Khwarizmi Eng. J.*, vol. 18, no. 1, pp. 38–50, 2022, doi: 10.22153/kej.2022.03.002.
- [2] Y. G. Khidhir and A. H. Morad, "Comparative Transfer Learning Models for End-to-End Self-Driving Car," *Al-Khwarizmi Eng. J.*, vol. 18, no. 4, pp. 45–59, 2022, doi: 10.22153/kej.2022.09.003.
- [3] M. K. Ahuja and A. Singh, "Static vision based Hand Gesture recognition using principal component analysis," *Proc. 2015 IEEE 3rd Int. Conf. MOOCs, Innov. Technol. Educ. MITE 2015*, pp. 402–406, 2016, doi: 10.1109/MITE.2015.7375353.
- [4] R. Kramer, C. Majidi, R. Sahai, R. J. Wood, and R. K. Kramer, "Soft curvature sensors for joint angle proprioception Printed-Circuit MEMS Fabrication of Sensors View project Soft Curvature Sensors for Joint Angle Proprioception," pp. 1919–1926, 2011, [Online]. Available: <https://www.researchgate.net/publication/221064831>.
- [5] E. Jespersen and M. R. Neuman, "Thin film strain gauge angular displacement sensor for measuring finger joint angles," *IEEE/Engineering Med. Biol. Soc. Annu. Conf.*, vol. 10, no. pt2, p. 807, 1988, doi: 10.1109/iembs.1988.95058.
- [6] E. Fujiwara, M. F. M. Dos Santos, and C. K. Suzuki, "Flexible optical fiber bending transducer for application in glove-based sensors," *IEEE Sens. J.*, vol. 14, no. 10, pp. 3631–3636, 2014, doi: 10.1109/JSEN.2014.2330998.
- [7] S. B. Shrote, M. Deshpande, P. Deshmukh, and S. Mathapati, "Assistive Translator for Deaf & Dumb People," *Int. J. Electron. Commun. Comput. Eng.*, vol. 5, no. 4, pp. 86–89, 2014.
- [8] H. P. Gupta, H. S. Chudgar, S. Mukherjee, T. Dutta, and K. Sharma, "A Continuous Hand Gestures Recognition Technique for Human-Machine Interaction Using Accelerometer and Gyroscope Sensors," *IEEE Sens. J.*, vol. 16, no. 16, pp. 6425–6432, 2016, doi: 10.1109/JSEN.2016.2581023.
- [9] M. Oudah, A. Al-Naji, and J. Chahl, "Hand Gesture Recognition Based on Computer Vision: A Review of Techniques," *J. Imaging*, vol. 6, no. 8, 2020, doi: 10.3390/JIMAGING6080073.
- [10] M. Al-Hammadi *et al.*, "Deep learning-based approach for sign language gesture recognition with efficient hand gesture representation," *IEEE Access*, vol. 8, pp. 192527–192542, 2020, doi: 10.1109/ACCESS.2020.3032140.
- [11] Y. S. Tan, K. M. Lim, and C. P. Lee, "Hand gesture recognition via enhanced densely connected convolutional neural network," *Expert Syst. Appl.*, vol. 175, no. November 2020, p. 114797, 2021, doi: 10.1016/j.eswa.2021.114797.
- [12] Prasad, A. K. ., M, D. K. ., Macedo, V. D. J. ., Mohan, B. R. ., & N, A. P. . (2023). Machine Learning Approach for Prediction of the Online User Intention for a Product Purchase. *International Journal on Recent and Innovation Trends in Computing and Communication*, 11(1s), 43–51. <https://doi.org/10.17762/ijritcc.v11i1s.5992>
- [13] A. G. Mahmoud, A. M. Hasan, and N. M. Hassan, "Convolutional neural networks framework for human hand gesture recognition," *Bull. Electr. Eng. Informatics*, vol. 10, no. 4, pp. 2223–2230, 2021, doi: 10.11591/EEI.V10I4.2926.
- [14] H. Y. Chung, Y. L. Chung, and W. F. Tsai, "An efficient hand gesture recognition system based on deep CNN," *Proc. IEEE Int. Conf. Ind. Technol.*, vol. 2019-Febru, pp. 853–858, 2019, doi: 10.1109/ICIT.2019.8755038
- [15] T. B. Waskito, S. Sumaryo, and C. Setianingsih, "Wheeled Robot Control with Hand Gesture based on Image Processing," *Proc. - 2020 IEEE Int. Conf. Ind. 4.0, Artif. Intell. Commun. Technol. IAICT 2020*, pp. 48–54, 2020, doi: 10.1109/IAICT50021.2020.9172032.
- [16] P. N. Huu, Q. T. Minh, and H. L. The, "An ANN-based gesture recognition algorithm for smart-home applications," *KSII Trans. Internet Inf. Syst.*, vol. 14, no. 5, pp. 1967–1983, 2020, doi: 10.3837/tiis.2020.05.006.
- [17] Wiling, B. (2021). Locust Genetic Image Processing Classification Model-Based Brain Tumor Classification in MRI Images for Early Diagnosis. *Machine Learning Applications in Engineering Education and Management*, 1(1), 19–23. Retrieved from <http://yashikajournals.com/index.php/mlaem/article/view/6>
- [18] F. Zhang *et al.*, "MediaPipe Hands: On-device Real-time Hand Tracking," 2020, [Online]. Available: <http://arxiv.org/abs/2006.10214>.
- [19] D. A. Taban, A. Al-Zuky, S. H. Kafi, A. H. Al-Saleh, and H. J. Mohamad, "Smart Electronic Switching (ON/OFF) System Based on Real-time Detection of

Hand Location in the Video Frames,” *J. Phys. Conf. Ser.*, vol. 1963, no. 1, 2021, doi: 10.1088/1742-6596/1963/1/012002.

- [20] B. J. Boruah, A. K. Talukdar, and K. K. Sarma, “Development of a Learning-aid tool using Hand Gesture Based Human Computer Interaction System,” *2021 Adv. Commun. Technol. Signal Process. ACTS 2021*, pp. 2–6, 2021, doi: 10.1109/ACTS53447.2021.9708354.
- [21] “Raspberry Pi OS.” <https://www.raspberrypi.com/software/> (accessed Dec. 10, 2022)
- [22] GitHub, “MediaPipe on GitHub.” <https://google.github.io/mediapipe/solutions/hands> (accessed Sep. 20, 2022).
- [23] R. E. Valentin Bazarevsky and Fan Zhang, “On-Device, Real-Time Hand Tracking with MediaPipe,” *MONDAY, AUGUST 19, 2019*. <https://ai.googleblog.com/2019/08/on-device-real-time-hand-tracking-with.html> (accessed Oct. 01, 2022).
- [24] C. Lugaresiet *al.*, “MediaPipe: A Framework for Perceiving and Processing Reality,” *Google Res.*, pp. 1–4, 2019.