Volume 07, Issue 01 (January-February 2024), PP 59-67 www.ijmsdr.org
ISSN: 2581-902X

International Journal of Medical Science and Dental Research

Wireless Music Control Using Gesture Implementation

Yash Datar[1], Reetu Jain[2]
[1](Student, Singapore International School, Mumbai, India)
[2](Chief Mentor, On My Own Technology, Mumbai, India)

Abstract: *This research article focuses on how gesture recognition technology can be used in wireless music controllers to improve accessibility for people with visual impairments. With over 7 million blind or visually impaired persons in the United States alone and an estimated 2.2 billion people worldwide suffering from vision impairment or blindness, there is an urgent need for creative solutions to raise the accessibility and quality of life for those who are visually impaired.*

The subject of gesture detection research is booming as a result of developments in game apps and other technologies for smartphones, computers, notebooks, and other portable devices. Originally, the gesture recognition system was built around a wearable hand glove-based sensor, but the cost of production was too high. Sensors based on camera vision are employed nowadays because of their simplicity and convenience of use. Any specific hand, facial, or body pattern or movement used to convey meaning in sign language is referred to as a gesture. Additionally, gestures can be expressed by the placement and combination of still hands. Multiple hand positions can be used to illustrate a continuous motion, whereas one hand position might be used to suggest a stance.

To be more specific, the research focuses on volume control by hand gestures using MediaPipe and OpenCV, which is consistent with the broader objective of gesture recognition technology. Using the webcam camera to interpret video, the process involves recognizing hand gestures, particularly those made with the tip of the index finger, and dynamically altering the device's volume in response. The project achieves its objectives with the use of crucial libraries such as MediaPipe, OpenCV, Pycaw, and NumPy. In order to adjust the volume, the camera on the device must first recognize hand landmarks, namely the points that represent the base of the palm and the tip of the index finger. The distance between these points is then used as a proportional measure.

Keywords -Hand Gesture Recognition, Mediapipe, Human-Computer Interface, Wireless Volume Control.

I. INTRODUCTION

Globally, 2.2 billion individuals are estimated by the World Health Organization to be blind or visually impaired. Of them, 1 billion have some sort of visual impairment that was preventable or remains untreated. Over 7 million people worldwide are blind or visually impaired, according to the National Federation of the Blind. These numbers show how significant visual impairments are on a global scale and how urgently innovative solutions are needed to improve accessibility and the standard of living for the blind and visually impaired. Gesture recognition technologies could make technology more accessible and easier to use for people with visual impairments when they are included into devices like wireless audio controls.

Using MediaPipe and OpenCV, the project allows volume control by hand gestures and is part of a broader gesture recognition community. Using the webcam camera, the primary objective of this project is to interpret video, recognize hand motions (particularly those involving the tip of the index finger), and dynamically modify the device's volume in response. The recommended volume control project uses many important libraries, including as OpenCV, MediaPipe, Pycaw, and NumPy, to achieve its goals. For the system to work, the camera needs to be able to identify hand landmarks, which are the locations that represent the base of the palm and the tip of that index finger. Then, by measuring the separation between these points, the volume of the gadget is modified accordingly.

Other aspects, like landmark identification, distance computation, mapping to volume range, and visual feedback display, are also described in the approach to emphasize the connection between hand gestures and volume control. A detailed description of the required installations and a step-by-step implementation guide are provided in the code details for the volume control project. It uses Pycaw for volume control, OpenCV for video capture, and MediaPipe for hand recognition. A box enclosing the index finger, hand landmarks, circles on finger tips, and a volume bar comprise the visual input that shows on the screen, showcasing the inventiveness of the code.

Overall, this research paper shows how gesture recognition technology can be applied in real-world scenarios including interactive volume control and sign language interpretation. The use of widely used libraries such as MediaPipe demonstrates the versatility of these technologies for a variety of creative and pragmatic purposes.

.

Fig 1. Gesture Recognition

II. Motivation and Novelty

The urgency of enabling visually impaired people to engage with technology in an accessible and natural manner serves as the driving force behind this initiative. Meeting this need in a novel and practical method is to employ the webcam camera for hand gesture detection and device control. The lives of persons who are blind or visually handicapped may be significantly impacted by this creative approach.

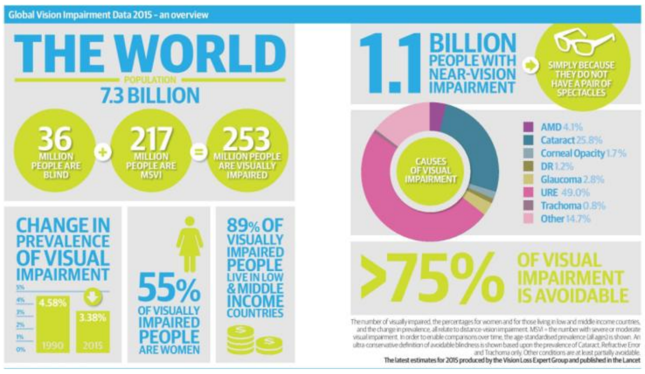


Fig 2. World Vision Impairment Statistics

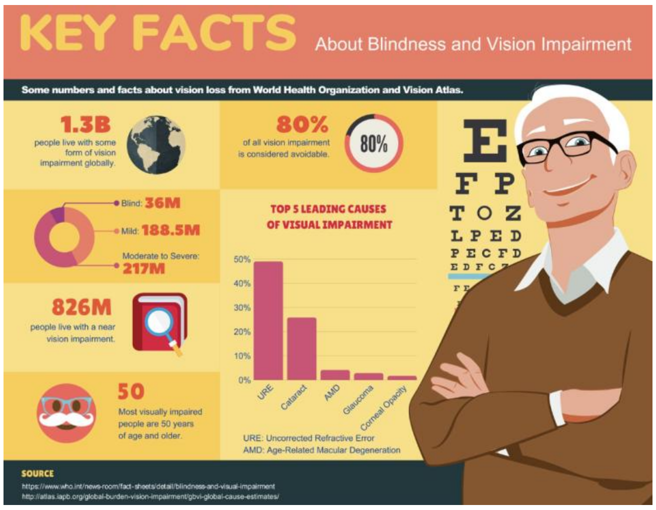


Fig 3. Key Facts About Blindness

The project's focus on hand gestures for wireless music control is a notable advancement in HCI, particularly in terms of accessibility for people with visual impairments. The innovative use of gesture recognition technology and the integration of common libraries draw attention to the project's distinctiveness and demonstrate its potential to favorably impact the accessibility needs of individuals with visual impairments. By concentrating on using necessary libraries such as MediaPipe, Open CV, Pycaw, and NumPy to achieve the project objectives, the method highlights the innovative combination of current technologies to create a unique wireless music control system. Furthermore, the innovative aspects of the project, such the visual feedback that shows up on the screen, demonstrate how deftly gesture detection technology can be applied to improve usability and accessibility.

III. LITERATURE REVIEW

Gesture recognition was a unique kind of human-computer interaction, first proposed by Myron W. Krueger in the mid-1970s. Given the recent and quick advances in both domains, it is now evident that computer hardware and vision systems are significant study areas. In their study [1], A. A. Abdulhussein and F. A. Raheem used grayscale photographs and edge detection techniques to identify hand gestures. Grayscale images are only 2D data, so there are limitations to this method that make it difficult to extract the key aspects of the hand.

Teak-Wei Chong and Boon-Giin Lee's research [2] offered a pricey and additional hardware-required way for identifying hand movements using a jump motion controller. Matteo Rinalduzzi's work offered a technique for identifying different hand motions with gloves and several hand sensors [3]. The motion detectors are pricey, and the hand finds it difficult to move due to the weight of the sensors. In order to identify ASL motions, Rajesh George Rajan and Dr. M. Judith Leo [4] discussed a deep learning technique that makes use of a public Kaggle dataset. Using the dataset, it applies several machine learning algorithms and gives a classified report on the results of each approach.

In the work by Aashni Hariaa, et al. [12], color features and contour extraction were used to identify the hand movements. The drawback is that because contour extraction is employed, it might be used to count the number of fingers sown rather than the different hand actions. A CNN-based technique for real-time hand gesture recognition in human-computer interaction was presented in Pei Xu's research [14]. The drawback is that while it can identify some hand signs used in human-computer interfaces, it cannot identify any motions used in sign language. In their study, Abhishek Bet al. suggested a system to recognize hand gestures using 3D CNN [16]. However, the process of recognizing motion-based gestures has not been covered.

In the paper by Gongfa Li et al. [17] provided instructions on how to recognize hand motions using CNN and Kinect. One drawback of Kinect is its high cost. The paper [18] by NoorkholisLuthfil Hakim et al. described a technique for recognizing dynamic hand gestures using LSTM and 3DCNN. The issue with this is that, rather of identifying motions for the alphabet, they employ them for military gestures.

In their study [5], Sourav Bhowmick, Sushant Kumar, and Anurag Kumar described how to use Deep Learning to recognize ASL gestures for two-letter combinations. The method's drawback is that it can only be applied to static photos, which makes it difficult to track motion-based movements. Gestures involving motion are not reliably detected.

In order to detect the gestures, OnamonPinsanoh, YuttanaKitjaidure, and Ariya Thongtawee's research [7] divided the frame into its left and right halves. Nevertheless, this process adds to the difficulty of gesture feature extraction. Chinmaya R. Naguri, et al. [8] described a technique for recognizing dynamic hand gestures from 3D motion in their work. This system is run with a Leap motion controller. Hand gesture detection and hand tracking with hand region segmentation in HCI are demonstrated in the paper by Qieshi Zhang, Jun Cheng, Jianming Liu, Kai Li, and Jianming Liu [9]. They utilized it for hand tracking and combined it with hand gesture recognition.

Examples of real-time speech creation and sign language recognition were presented in Thakur et al.'s research [25]. They used the CNN approach in the paper. The moving indicators have been replaced by still graphics, which is a drawback. The background study demonstrates that new methods are required to fully exploit the range of innovative computer vision algorithms and that more work hasto be done in the field of gesture identification.

IV. PROPOSED METHODOLOGY

The process entails using the webcam camera to analyze video, recognizing hand gestures (particularly those performed with the index finger), and dynamically varying the device's volume. Key libraries including OpenCV, MediaPipe, Pycaw, and NumPy are used to achieve the project's objectives. The camera recognizes hand landmarks, which are the points that represent the base of the palm and the tip of the index finger. It then adjusts the volume of the gadget by measuring the distance between these spots proportionately. This is the process at work.

V. Media pipe

Mediapipe hand key point detection is driven by a machine learning pipeline. A palm identification model that examines the full image is one of the models in the MediaPipe machine learning pipeline. After processing the entire image, it produces an orientated hand-bounding box. a hand landmark model that uses the picture region cut by the palm detector to generate incredibly accurate 3D hand keypoints[15].

Moreover, the machine learning pipeline uses the hand landmarks found in the previous frame to crop the hand, and palm detection is only used to localize the hand in cases where the land-mark model is unable to do so. Figure 4 shows the 21 hand landmarks that can be tracked with the Mediapipe's hand landmark detector.

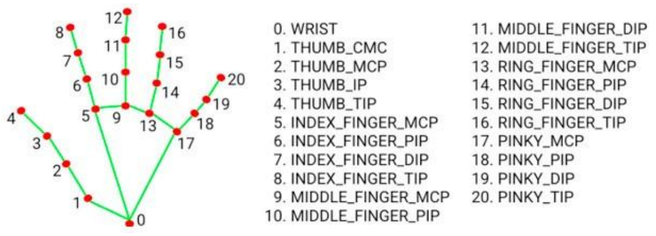


Fig 4. Hand Landmarks Representation used in Mediapipe

Arsheldy Alvin, et al [13] have used Mediapipe and K nearest neighbours algorithm to determine the hand gestures which helps us in understanding how to use mediapipe for hand gesture recognition.

The main distinction between Mediapipe and Tensorflow is that the former is an artificial intelligence and machine learning software library, while the latter is not. Though its main focus is on deep neural network training and inference, it can be applied to a wide range of tasks. For live and streaming media, on the other hand, Medipipe offers cross-platform, freely available machine learning solutions that are customizable. This work utilizes mediapipe to record the hand keypoints and Tensorflow to train and identify the machine learning algorithm. Both the CPU and GPU are used by Medipipe in its operations. More processing power is not needed for Medipipe to function.

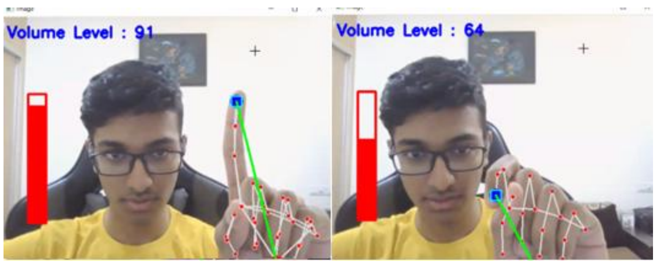


Fig 5. Video Capture of MediaPipe Landmarks

VI. Hand Gesture Recognition Using Pycaw

The hand gesture recognition project aims to provide a natural and dynamic way to use hand gestures to adjust the volume on a device. For hand gesture recognition and volume control, the project effectively utilizes modules such as Pycaw, MediaPipe, and NumPy. The MediaPipe Hands library, which is used to infer 21 crucial 3D hand properties from a single frame, allows for the precise identification of hand landmarks. Using the camera to detect hand landmarks and calculate the distance between the tips of the index and palm fingers is the fundamental concept behind this project. After that, the volume range is connected to this distance, allowing for dynamic volume control in reaction to hand movements. The code uses MediaPipe Hands to recognize hand landmarks, Pycaw to control the system volume, and OpenCV to capture webcam video. MediaPipe Hands is used to locate hand landmarks in the collected frame after processing and RGB conversion.

The mapping of volume levels is carried out by taking the measured distance between specific hand markers. The processed video frame is displayed along with visual feedback features like hand landmarks, circles on finger tips, a box around the index finger, and a volume meter. The project includes an exit command ('Spacebar') for user convenience. All things considered, the project provides a novel and interesting way to control a device's volume with hand movements.

VII. Code Implementation

With the available Python version, controlling the system volume with hand gestures is made interesting and engaging. The code must execute numerous pip install instructions in order to build up the necessary libraries, which include OpenCV (cv2), MediaPipe, Pycaw, and NumPy. These libraries are necessary for processing video frames, identifying hand landmarks, and controlling system volume. The code starts by using the webcam to take a photo, converting it to RGB, and then transferring it to the MediaPipe Hands library for processing. Next, hand motions are identified using the landmark list returned by the MediaPipe Hands library for each hand in the frame.

The code iterates through the list of landmarks and draws circles at the tips of the index and palm fingers to provide the user with visual feedback. A line is also drawn between the index finger and palm base to help measure the distance between them. Then, by measuring the distance between the tips of the palm base and index finger, the system volume is adjusted. The following formula is used by the code to convert the distance to a volume level:

volume = np.interp(length, [30, 350], [volMin, volMax])

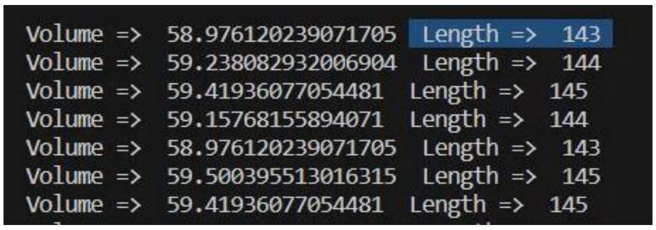
The minimum and highest volume levels are indicated by the *volMin* and *volMax* variables, respectively, while the length variable shows the separation between the points of the palm base and index finger. This formula, which maps the distance to the system volume range, can be used to regulate the volume dynamically with hand gestures. The code then sets the system volume level to the computed value, adjusting the volume in reaction to the user's hand movements. The user is given visual clues indicating the current volume level by the code, which displays the video frame together with the generated circles, line, and volume bar.

All things considered, this Python implementation offers an interesting and dynamic way to control system volume with hand movements, showcasing the inventive application of gesture detection technologies. This project can be expanded to include additional gesture-detecting applications, such as navigating presentations or managing music playback. Additionally, the implementation can be enhanced for real-time speed, making it suitable for use in a range of contexts, such as virtual reality and gaming.

In conclusion, anyone interested in learning more about gesture detection technologies can start with this code. It is easy to use, flexible to fit different needs and scenarios, and well-documented. The study additionally demonstrates how the integration of diverse libraries and technologies may yield innovative solutions that enhance user engagement and experience.

VIII. Results

The suggested approach makes use of webcam technology to dynamically adjust the loudness of the device and recognize particular hand gestures, especially the tip of the index finger. For this, NumPy, Pycaw, MediaPipe, and OpenCV are used. The project entails identifying hand landmarks, particularly the tip of the index finger and the base of the palm, and adjusting the device's loudness based on the distance between these locations. In order to regulate the system volume, the code records webcam frames, runs them through MediaPipe Hands, and determines the separation between hand landmarks. This creative solution illustrates how gesture recognition technology may be used to improve user experience by providing an interactive way to manage system loudness using hand motions.



```
Volume =>  58.976120239071705  Length =>  143
Volume =>  59.238082932006904  Length =>  144
Volume =>  59.41936077054481  Length =>  145
Volume =>  59.15768155894071  Length =>  144
Volume =>  58.976120239071705  Length =>  143
Volume =>  59.500395513016315  Length =>  145
Volume =>  59.41936077054481  Length =>  145
```

Fig 6. Volume versus Distance Calculation

IX. Experimental results

Using the MediaPipe, Pycaw, and NumPy libraries, the suggested methodology was successfully implemented to enable effective hand gesture recognition and volume control. Through the accurate detection of 21 crucial 3D hand landmarks from the webcam feed, the project was able to precisely modulate volume based on hand gestures by utilizing the MediaPipe Hands library. The algorithm effectively processed video frames, recognized hand landmarks, and calculated the distance between the base of the palm and the tip of the index finger in order to dynamically adjust the system volume. The study showcased how gesture recognition technology may enhance user experience by demonstrating an interactive method for manipulating system loudness with hand gestures. The application demonstrated how several libraries and technologies may be effectively combined to provide an innovative solution that improves user interaction.

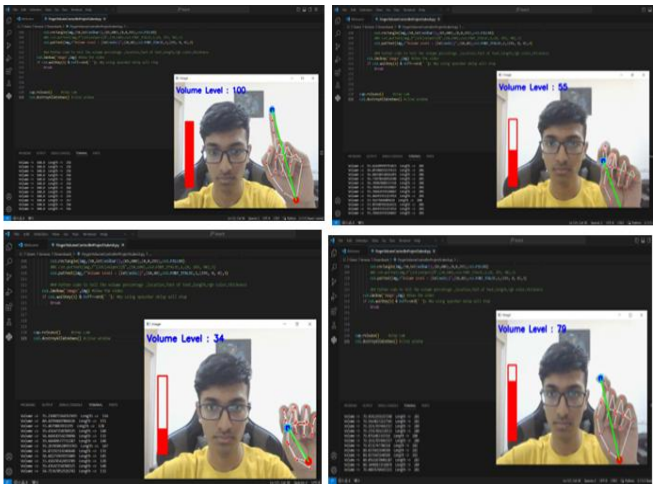


Fig 7. Four Test Results of Volume Control

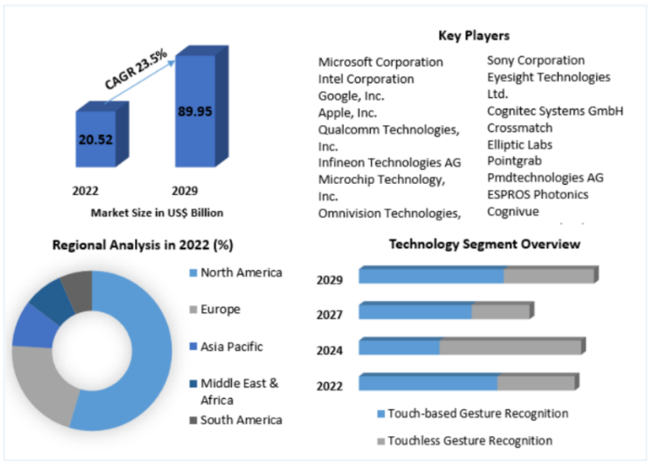


Fig 8. Gesture Recognition and Touchless Sensing Market (2022 to 2029)

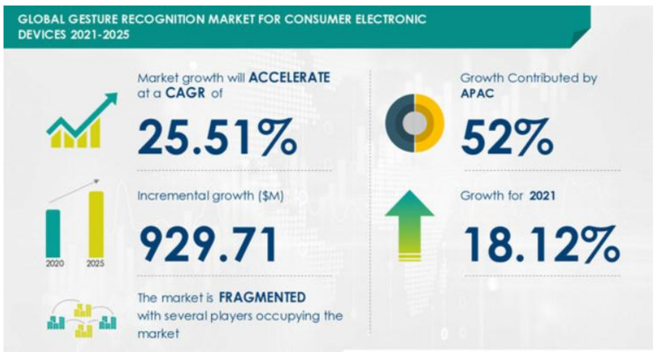


Fig 9. Growth Statistics of Gesture Recognition Market for Consumer Electronic Devices

X. CONCLUSION

In conclusion, this study demonstrates the adaptability and value of gesture recognition technology in a range of settings. The creative application of well-known tools such as MediaPipe demonstrates their versatility in providing practical and interactive solutions. Moreover, this technology holds immense promise for enhancing accessibility, particularly for individuals with disabilities, and promoting greater inclusivity in technology as a whole. Additional proof of the importance of these innovative and inclusive technological advancements comes from statistics demonstrating the prevalence of disability in society.

This research focuses on hand gestures for volume control using MediaPipe and OpenCV to provide new uses for gesture detection technology. Essentially, the idea is to employ webcams to identify and interpret hand gestures, with an emphasis on the index finger tip, in order to enhance user-device interaction. A methodical approach is provided by the methodology, which incorporates key libraries such as MediaPipe, OpenCV, Pycaw, and NumPy.

The process is based on the camera's recognition of hand landmarks, specifically the tips of the index and palm fingers. The distance between these landmarks offers a proportionate means of adjusting the device's volume dynamically. The project's methodology, which emphasizes the smooth integration of hand gestures and volume control, consists of mapping to a volume range, distance calculation, landmark identification, and visual feedback presentation.

Fig 10. Future Automation Enhancements of Gesture Recognition

Implementation details for the project include a comprehensive how-to that highlights the use of Pycaw for volume control, MediaPipe for hand detection, and OpenCV for video capture. A box enclosing the index finger, hand landmarks, fingertip circles, and a volume meter are among the visual cues that the code comprises that amply illustrate its inventiveness.

REFERENCES

[1] A. A. Abdulhussein and F. A. Raheem, "Hand gesture recognition of static letters American sign language (ASL) using deep learning," Engineering and Technology Journal, Vol. 38, No. 06, pp. 926-937, 2020.

[2] Teak-Wei Chong and Boon-Giin Lee, American Sign Language Recognition Using Leap Motion Controller with Machine Learning Approach, Department of Electronics Engineering, Keimyung University, Daegu 42601, Korea, October 2018.

[3] Matteo Rinalduzzi, Alessio De Angelis, Francesco Santoni, Emanuele Buchicchio, Antonio Moschitta, Paolo Carbone, Paolo Bellitti, Mauro Serpelloni, Gesture Recognition of Sign Language Alphabet Using a Magnetic Positioning System, June 2021.

[4] Rajesh George Rajan, Dr.M.Judith Leo, American Sign Language Alphabets Recognition using Hand Crafted and Deep Learning Features IEEE Xplore Part Number:CFP20F70-ART published on 2020.

[5] Sourav Bhowmick, Sushant Kumar and Anurag Kumar, Hand Gesture Recognition of English Alphabets using Artificial Neural Network, published on 2015.

[6] Ross E. Mitchell, Gallaudet Research Institute, Draft manuscript accepted for publication in Sign Language Studies, Volume 6, Number 3, 2006.