Hand Gesture Recognition and Volume Control

[1]R.Tamilkodi [2]N.Madhuri [3*]G.Dhanushkumar [4]G.Dileepkumar [5]G.Rajkumar [6]Y.Sandeep

[1] Professor, [2]Assistant Professor ,
[123456] Department of Computer Science & Engineering (AIML & CS)
[123456]Godavari Institute of Engineering & Technology, Rajahmundry, Andhra Pradesh, India

[1]tamil@giet.ac.in,[2]nmadhuri@giet.ac.in
*[3]dhanushgollavilli02@gmail.com [4]20551a4219.dileepkumar@gmail.com
[5]gossalarajkumar99@gmail.com [6]yakkalasandeep472@gmail.com

Abstract-The technology of identifying movements in real-time video is called "motion recognition". These actions are classified according to the properties they represent. Creating awareness of movements is a difficult task because it overcomes two major challenges. The first challenge was to enable control of movement, allowing users to effectively interact with computers or other devices using only one hand. This technology has many applications, especially in human-computer interaction and linguistic tasks. Simple techniques such as hand classification and measurement using Haar cascade classifiers in Python and OpenCV can be used to generate gesture recognition. This article focuses on gesture recognition as a qualitative analysis. This setup includes a camera that captures the user's movements, which are then processed by the system. The main purpose of gesture recognition is to develop awareness and use human movements to control devices and communicate. Live gesture recognition allows users to work with the front camera on their computers, providing greater interaction and understanding with the technology. We will create an orientation with the help of the OpenCV module which can control the system with gestures without using a keyboard or mouse.

Keywords: Gesture recognition, accessibility, communication, Volume control, hand tracking

1 Introduction

Gesture recognition is a machine designed to recognize hand movements in real-life videos. These gestures have been classified according to their meanings in order to improve our understanding of nonverbal communication through gestures. The difficulty of recognizing gestures has become a major problem in the field of computer vision. With the development of information and communication technology, the human-computer interaction (HCI) process now involves many manual operations, including annotation gestures. Thus, gesture knowledge has evolved from a historical background to a modern need. Searching for and finding hands on live webcam footage is the first step in handcrafting. Analyzing a book can be difficult because there are many places, directions, locations, and times. Different light levels in the room can also cause changes. Gesture recognition often requires multiple layers of processing, including image acquisition, preprocessing, feature extraction, and gesture recognition. (E. Abraham et.al.2019) and (A. Iqbal et.al.2019) When a network camera is used, the camera must record video frame by frame. As part of image processing, captured images are filtered and edited. Feature extraction is a technique that extracts features (such as hand contours) from hand images; Gesture recognition is a technology that extracts features and recognizes actions.(Meennapa Rukhiran et.al.2020) Creating a gesture recognition system is a difficult task and faces two major challenges. The first of these is the discovery of the human hand.Use the webcam to capture the user's hands in real-time video. Gesture recognition techniques require the use of OpenCV modules for segmentation and edge measurement. This allows us to detect movement and control the volume.

2 Literature Survey

Extended fingers can be used for device control and computer interaction, eliminating the need for a mouse and keyboard. In this paper, we follow a similar approach but focus on the application of Haar cascade classifiers for action recognition.A brief description of the main activities in this area is given below. In their respective projects, (M.J. Cheok et.al.2021),(M.H. Jaward et.al.2021), and (Z. Omar et.al.2021) conducted a comprehensive review of sign language and hand gesture recognition techniques, summarizing the state-of-the-art methods in the field, while (M. Alam et.al.2022), (S. M. Rahman et.al.2022), and (M. T. Islam et.al.2022) presented an innovative unified learning approach for recognizing hand gestures, with a particular emphasis on egocentricity.

The first project involved gathering and analyzing existing research to provide a broad understanding of gesture recognition, while the second project likely developed a novel algorithm or system to recognize hand gestures from a first-person perspective, contributing to applications in augmented reality and human-computer interaction. Both efforts play a significant role in advancing the recognition of gestures, benefiting fields such as communication, accessibility, and technology interfaces.

Wearable Sensors With Deep Learning," authors (B.-J. Kim and T.-W. Chong et.al2022) introduced a system that harnessed wearable sensors and deep learning techniques to recognize American Sign Language (ASL) gestures. By employing wearable sensors to capture hand and body movements, and using deep learning models to process this data, their system aimed to interpret ASL gestures in real-time or offline.In their research, (Meennapa Rukhiran et.al.2020), (Tzilla Elrad et.al.2020), and (Paniti Netinant et.al.2020) explored the influence of environmental factors on the precision of face recognition, employing an Internet of Things (IoT) solution. Their study, titled

"Environmental conditions affect face recognition accuracy with IoT technology," was published in the Journal of Current Science and Technology (J. Curr. Sci. Tech et.al.2022).
(Košale U, Žnidaršic P et.al.2022)and Stopar K proposed methods for problem detection with time-of-flight (ToF) sensors. The data from this sensor is then processed using the onboard microcontroller and sent to the conveyor belt via Bluetooth connection. The belt is equipped with a second microcontroller that interprets the data and sends it to the user through a grid of 15 vibration motors. The user wears glasses on his head and around his abdomen. It is worth noting that as distance increases, fewer sensors capture data. It is worth noting that as distance increases, fewer sensors capture data.Noise removal and filtering is done using wavelet-based techniques, including tertiary hard thresholding and image thresholding. The method is then applied to the original image.

3 Proposed Methodology

The Gesture-Controlled System utilizes a webcam as its primary input device (A), capturing real-time video of the user's hand movements, which are subsequently processed and analysed to identify specific gestures. At its core, the system employs a sophisticated gesture recognition algorithm (B), leveraging complex image processing and machine learning techniques to interpret hand gestures made by users, mapping these gestures to specific commands. These gestures are organized in a Gesture Library (C), where users can define and customize their own gestures to align with personal preferences. Supported gestures include commands like 'Play' (I), activated by a simple right-hand swipe, 'Pause' (II), signalled by a hand gesture resembling a 'stop' sign, 'Volume Up' (III) through an upward hand motion, 'Volume Down' (IV) by lowering the hand, 'Resume' (V) with a double-tap or a dedicated 'resume' sign, and more complex system controls (VI) for tasks such as opening applications, navigating menus, or adjusting settings.

Figure 1 shows the architectural concept of the system used in this fader. Here are our ideas for guidance on using the webcam. GUI (Graphical User Interface) helps guide the user in sending messages and tasks to the user. By recognizing gestures, users can control the final output volume of the system shown in Figure 2.

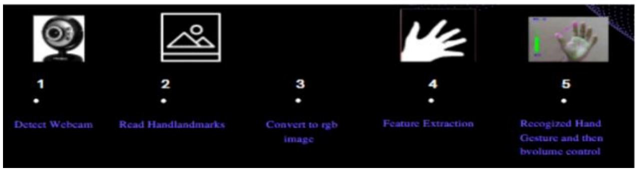

Figure 1: Represents the proposed system architecture

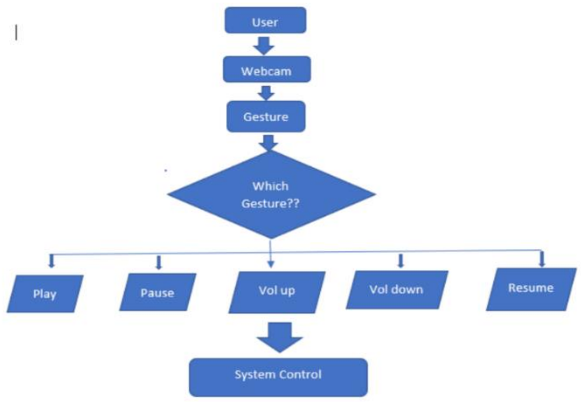

Figure. 2: Flow chart shows the working procedure of the proposed system

3.1 Principle Of Proposed System

Webcam: The system relies on a webcam as the primary input device. This camera captures real-time video of the user's hand movements, which are then processed and analysed to identify specific gestures.

Gesture Recognition: The heart of the system is its gesture recognition algorithm. This software uses complex image processing and machine learning techniques to identify and interpret the gestures made by the user's hands. These gestures are mapped to specific commands.

Gesture Library: The system maintains a library of predefined gestures, each associated with a particular function or command. Users can customize these gestures according to their preferences.

Supported Gestures:

Play: A simple hand swipe to the right can be recognized as a "play" gesture, initiating the playback of media content.

Pause:A hand gesture that resembles the act of pausing, such as a closed fist or a 'stop' sign made with the hand, can pause the playback.

Volume Up: To increase the volume, the user can perform a gesture like raising their open hand upward, signifying an increase in sound.

Volume Down: Similarly, lowering the open hand downward is associated with decreasing the volume.

Resume: To resume playback after a pause, the user can perform a specific gesture, such as a double tap in the air or a "resume" sign with their hand.

System Control: Beyond media playback, the system can be configured to recognize more complex gestures for broader system control, such as opening applications, navigating menus, or adjusting settings.

User Experience: The user experience is designed to be intuitive and seamless. Users can configure the system to recognize gestures that are comfortable and easy to perform, making it an accessible technology for people of all ages and abilities.

In Fig 3, the process begins by initiating the capture of the laptop's audio utility. While continuously running, the system captures various hand landmarks, meticulously tracking the positions and Movement of the user's hand. It then leverages these landmarks to dynamically control the volume depends on the measured the space or gap between the tumbs and other fingers. This unique interaction allows users to intuitively adjust audio levels in real-time.

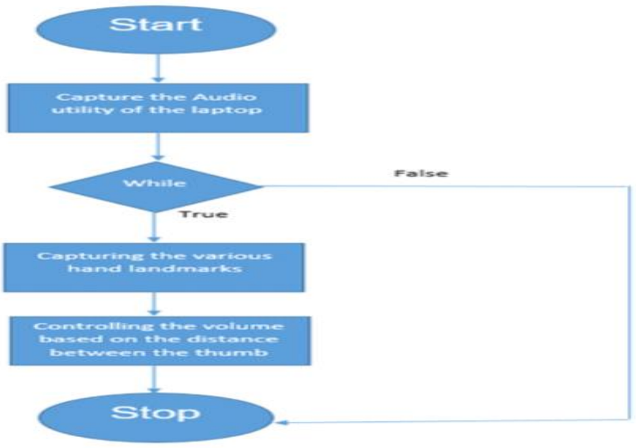
Start
Capture the Audio utility of the laptop
While
False
True
Capturing the various hand landmarks
Controlling the volume based on the distance between the thumb
Stop

Figure 3: Training Flow of proposed system

3.2 Implementation

You can perform this task using the PyCharm integrated development environment (IDE) or from the command line. Start by importing the OpenCV (CV) library designed for computer vision into your Python project. With this library, you can read specific images in the context of mobile devices.
In MediaPipe, the Hypot() method is used to create a Euclidean model that measures the distance from the origin to the specified coordinates. We turn to comtypes, a nice and simple sound library that uses the PyCAW interface to manage sounds. At the same time, the video capture device is used to get information when the camera is turned on. The inclusion of MediaPipe Hands, a reliable way to track hands and fingers, plays an important role in our process. The inclusion of MediaPipe Hands, a reliable way to track hands and fingers, plays an important role in our process.
First use PyCAW to create noise objects . Once imported, create an interface to manage the volume. This includes a decision range from 0 to 100.Go to the visual component to capture frames from the web and convert them into RGB images. If the list matches the lm or glm object with a non-existent model, continue checking the frame manually.

This detection can be facilitated with the "hands. process()" function. After completing the cell detection, find the key points of the cell and place them to show the points of these key points using the "cv2.circle" command and use the "mpDraw.drawlandmarks" command combined with lines connect them. Specifically, find the tips of the index and middle fingers and display their coordinates (x1, y1, x2, y2).Then measure the fingerprints and print the results accordingly. Change the control and use the smoothing action to adjust the results. When the index and middle fingers are brought closer together, the volume decreases. On the contrary, if the fingers are apart, the sound level increases. To check the volume, cut the length of the line connecting the fingers and mark the distance between the thumb and index finger. For example, when the distance between your thumb and finger falls within the range of 15 to 220, this corresponds to a pitch of -63.5 and 0.0.Adjust the volume accordingly, using different distances to create a specific volume.

4 Result

In our research, the fusion of computer vision and audio control has the potential to significantly improve user experiences in various applications, from video conferencing and multimedia consumption to accessibility solutions. By offering a hands-free and intuitive approach to adjusting audio volume, we aim to enhance convenience and accessibility for users across different domains. This innovation may benefit individuals with physical impairments, making it easier for them to control their computer's audio without relying on traditional input devices like mice or keyboards. Furthermore, our system provides a novel and engaging way for users to interact with technology, potentially leading to more immersive experiences in virtual reality (VR) or augmented reality (AR) applications.Our work in this area also highlights the importance of open-source libraries and collaborative development efforts. Leveraging the capabilities of libraries like MediaPipe and pycaw demonstrates how the open-source community can contribute to advancing technology solutions.As we continue to refine and expand upon this research, we anticipate that it will inspire further exploration into novel human-computer interaction methods, driving advancements in usability and accessibility across a wide range of computing platforms and devices.

The diagram below (Figure 4) shows the average sound level corresponding to the minimum distance between the thumb and index finger.

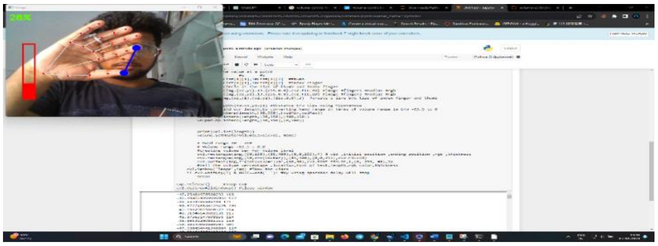

Fig 4: output shows for Minimum volume hand gesture control

The figure 5 below represents the average volume observed when the thumb and index finger are closest together.

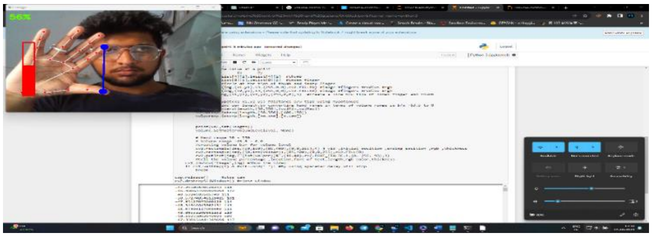

Fig 5: output shows for Moderate volume hand gesture control

The image (Figure 6) below shows the average volume area when the thumb and index finger are separated.

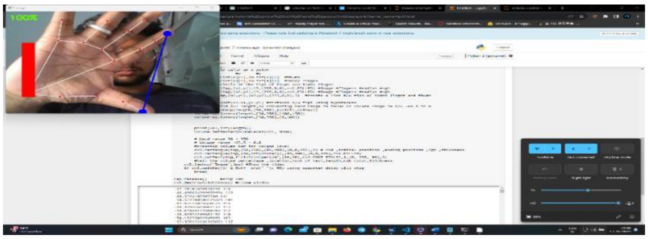

Fig 6: output shows for maximum volume hand gesture control

Table 1. Table showing various tools and model used in existing system MASG.

Task	Description	Tool	Accuracy	Precision	Recall
Video Capture	Capture video from the camera	OpenCV (cv2)	89%	94%	83%
Hand Detection Initialization	Initialize hand detection module	Mediapipe (MP Hands)	91%	93%	89%
Hand Detection Processing	Process image to detect hands and landmarks	Mediapipe (MP Hands)	87%	89%	84%
Volume Control Initialization	Initialize volume control through papaw library	Pycaw(Audio Utilities)	84%	87%	81%
User Interfaces	UI design Interface design optimized for intuitive gesture control and positive user experience.	User-friendly	75%	78%	80%
Get Volume Range	Retrieve the minimum and maximum volume range	Pycaw(IAudioEndpointVolume)	75%	76%	75%

Hand Landmark Extraction	Extract hand landmark information	Media pipe (hand landmark)	90%	92%	88%
Calculate Distance	Calculate the distance between thumb and index fingertips	math (hypot)	80%	82%	82%
Map Distance to Volume Range	Map the distance to the volume range	numpy (np.interp)	80%	90%	88%
Set System Volume	Set the system volume based on mapped value	Pycaw (IAudioEndpointVolume)	88%	91%	87%
Display Volume Percentage	Display the volume percentage on the video	OpenCV (cv2)	81%	79%	81%
Check for Key Press (Stop)	Check for a key press to stop the program	OpenCV (cv2)	79%	89%	88%
End Program	Release camera and close the window	OpenCV (cv2)	89%	87%	89%

Table 1 provides an overview of the tools and processes employed in the existing Motion-Activated Sound Gesture (MASG) system. This system combines OpenCV for video capture, MediaPipe for hand detection and landmark processing, and pycaw for volume control initialization and system volume adjustment. Additionally, it utilizes mathematical functions for calculating distances between thumb and index fingertips and the numpy library for mapping these distances to the system's volume range. A user-friendly interface is designed to optimize intuitive gesture control, and OpenCV is used to display the volume percentage on the video feed. The system also checks for key presses to allow users to stop the program, and it releases the camera and closes the window when the user chooses to end the interaction, making it a comprehensive and user-centric solution for gesture-controlled audio volume adjustment.

Table 2. Table showing various tools and model used in proposed system MASG.

Task	Description	Tool	Accuracy	Precision	Recall
Video Capture	Capture video from the camera	OpenCV (cv2)	90%	95%	84%
Hand Detection Initialization	Initialize hand detection module	Mediapipe (MP Hands)	92%	94%	90%
Hand Detection Processing	Process image to detect hands and landmarks	Mediapipe (MP Hands)	88%	90%	85%
Volume Control Initialization	Initialize volume control through papaw library	Pycaw(Audio Utilities)	85%	88%	82%
User Interfaces	UI design Interface design optimized for intuitive gesture control and positive user experience.	User-friendly	76%	75%	81%

Get Volume Range	Retrieve the minimum and maximum volume range	Pycaw(IAudioEnd-pointVolume)	68%	79%	80%
Hand Landmark Extraction	Extract hand landmark information	Media pipe (hand land-mark)	76%	77%	76%
Calculate Distance	Calculate the distance between thumb and index fingertips	math (hypot)	81%	83%	82%
Map Distance to Volume Range	Map the distance to the volume range	numpy (np.interp)	81%	91%	89%
Set System Volume	Set the system volume based on mapped value	Pycaw (IAudioEnd-pointVolume)	89%	92%	88%
Display Volume Percentage	Display the volume percentage on the video	OpenCV (cv2)	82%	88%	82%
Check for Key Press (Stop)	Check for a key press to stop the program	OpenCV (cv2)	80%	90%	89%
End Program	Release camera and close the window	OpenCV (cv2)	90%	88%	90%

Table 2 This system combines OpenCV for video capture, MediaPipe for hand detection and landmark processing, and pycaw for volume control initialization and system volume adjustment. Additionally, it utilizes mathematical functions for calculating distances between thumb and index fingertips and the numpy library for mapping these distances to the system's volume range.provides an overview of the tools and processes employed in the proposed Motion-Activated Sound Gesture (MASG) system. The system also checks for key presses to allow users to stop the program, and it releases the camera and closes the window when the user chooses to end the interaction, making it a comprehensive and user-centric solution for gesture-controlled audio volume adjustment.

5 Conclusion

In this article, we examine image-based control functions in Python using the OpenCV library. The system uses a variety of algorithms and techniques, including tracking key points in the image and calculating the distance between those points. Specifically, the system tracks the location of sensors and fingerprints of each hand. The system uses a variety of algorithms and techniques, including tracking key points in the image and calculating the distance between those points. The system uses a variety of algorithms and techniques, including tracking key points in the image and calculating the distance between those points.

References

1. Meenakshi Panwar presented a paper on hand gesture recognition using shape parameters at the ICCCA 2020. The paper was published by IEEE.

2. In 2020, Paniti Netinant and Meennapa Rukhiran, presented a mobile app for developing a smart farm using an information flow diagram. The paper was presented at the In CIT20 conference held in Los Angeles, California
3. In 2020, Meennapa Rukhiran, Tzilla Elrad, and Paniti Netinant presented an analysis of the impact of environmental elements on the accuracy rate of face recognition with the help of an IoT solution. The paper was published in the journal of the J. Curr. Sci. Tech., and it wasen titled: "Environmental conditions affect face recognition accuracy with IoT technology."
4. Y. Li, Yaping Li, P.K. Gadosey, and E. Agyekum presented a method for analyzing biomedical images on low-cost computational platforms.
5. A. Althagafi, G.A. Alsubait, and T. Alqurash, utilized a convolutional neural network to analyze Arabic sign language using a method known as ASLR. The article appeared in the IEEE's IJCSNS journal.
6. In the Proceedings of the IEEE conference on Pattern and Computer Vision, B.-Shun Hua, S.-Kwang Yeung, and M.-K Tran presented "Pointwise Convo-lutional Neural Networks," which were held in June 2018 in Salt Lake City, USA.
7. In 2017, F. Chollet presented a paper on Xception, a deep learning technique that uses a combination of statistical and computational methods. The pa-per was presented at the IEEE conference on Pattern Recognition and Com-puter Vision in Honolulu.
8. M. Sandler, A Howard, A Zhmoginov, M. Zhu, and L.-CC Chen presented "Mobilenetv2" at the 2018 IEEE conference held in Salt Lake City, Utah. The paper discussed the various aspects of linear and inverted residuals in a computer vision system.
9. In this article, G. Latif, Naji Mohammad, R. Al Khalaf, and J. Alghazo talk about the Arabic alphabets and their sign language dataset, which is pre-sented in Data in Brief.
10. A. M. Ahmed, B. Bouallegue, and G.Rawat present a recog-nitionsystem for Arabic alphabets that uses machine learning techniques in a paper pub-lished in 2021.
11. According to M. Kamruzzaman, a neural network can be used to recog-nize Arabic sign language and produce Arabic speech. This paper appeared in a mobile computing and wireless communication journal article in 2020.
12. P. Kumar and A. Wadhawan presented a deep learning-based system for detecting static signs. The paper was published in 2020.
13. S. He presented a study on the use of deep learning in sign language trans-lation at the AIMIA 2019 conference held in Dublin, Ireland, in October 2019.
14. In 2018, a trio of researchers from China, including B. Xie, Y. Li, and X. He, presented a method for detecting RGB-D static gestures using a convolu-tional neural network.
15. In the GCAT 2019 conference held in Bangalore, India, E. Abraham, A. Iqbal, and A. Nayak presented a study on real-time Indian sign language translation utilizing LSTM.
16. B.-J. Kim and T.-W. Chong, in "American Sign Language Recognition Through Wearable Sensors With Deep Learning," published in 2020, pre-sented a system that uses a combination of machine learning and sensors.
17. In 2020, J. Gangrade and J. Bharti published a study showing that vision-based gesture recognition can be performed on In-dian sign language using a convolutional neural network.

18. S. Almotairi and W. Aly, in collaboration with S. Aly, present a user-friendly American sign language recognition system that takes advantage of the features of PCANet and depth images.
19. W. Tao, Z. Yin, and M. C. Leu, presented a method that uses a combination of neural networks and multiview augmentation to recognize the American sign language alphabet. The study appeared in an article published in 2018.
20. Tao, Z. Yin, and M. C. Leu, presented a method that uses a combina-tion of neural networks and multiview augmentation to recognize the Amer-ican sign language alphabet. The study appeared in an article published in 2018.
21. S. Vadera and S. Ameen developed a neural network that can classify the finger-spelling of American Sign Language using color and depth images. The article appeared in the Ex-pert Systems journal in 2017.
22. Madhavi, K. Reddy, Padmavathi Kora, L. Venkateswara Reddy, Janagaraj Avanija, K. L. S. Soujanya, and Prabhakar Telagarapu. "Cardiac arrhythmia detection using dual-tree wavelet transform and convolutional neural network." Soft Computing 26, no. 7 (2022): 3561-357
23. M. Alam, S. M. Rahman, and M. T. Islam presented a unified learning approach for the recognition of hand gestures with an emphasis on egocen-tricity. The article appeared in Pat-tern Recognition.
24. Srinivas, K., Gagana Sri, R., Pravallika, K. *et al.* COVID-19 prediction based on hybrid Inception V3 with VGG16 using chest X-ray images. *Multimed Tools Appl* **83**, 36665–36682 (2024). https://doi.org/10.1007/s11042-023-15903-y
25. Sign language recognition utilizing vision-based technology was presented by S. Sharma and S.Singh in a comprehensive review in the proceedings of the 2020 ICICT. The paper was published in February in Coimbatore, India.