ISSN No:-2456-2165

Hand Gesture Control System using Open CV

Nikita Chaudhari, Kaustubh Sonawane, Arathi Boyanapalli
Department of Computer Engineering,
SIES Graduate School of Technology, University of Mumbai, Nerul Navi Mumbai, India

Abstract:- All around the world, researchers are working to improve the interactivity of digital devices and to make them operate with the least amount of physical interaction possible. The interactive computer system proposed in this study can function without a keyboard or mouse. By the implementation of this project, everyone would have an advantage including a paralyzed person who finds it difficult to use a mouse or keyboard. The easiest and most natural way to communicate is through hand gestures and hence an innovative method of computer control using a real-time web camera is demonstrated in this Hand Gesture Control project. The approach is to use a camera and computer vision technologies to handle tasks and perform the jobs without the use of a physical mouse or keyboard. A webcam with normal resolution is used to track the user's hand in two dimensions. The system is created using Python and OpenCV. The output of the camera will be seen on the monitor and the tasks associated with the hand gestures will be performed. Thereby, an application of an easy-to-use gesture control system is shown in this project.

Keywords:- *Interactive Computer System, Hand Gesture, Webcam, Keyboard, Mouse.*

I. INTRODUCTION

Nowadays, gesture control applications are becoming an area of wide interest with evolving technology. Keyboards and mouse have been the primary input methods for computers for many years. However, as ubiquitous, and ambient technology (like PlayStations) and tools that let users hold virtual items become more and more common, hand or body gestures are becoming increasingly important. Gesture controllers are now an integral part of the human computer interaction system. Computers can grasp human body language with the help of gesture recognition. Instead of using only the simple text-user interfaces or graphical-user interfaces, the gesture recognition system aids in creating a stronger connection between humans and technology (GUIs). The gesture recognition is an advancement of Touch User Interface (TUI), a type of interaction between a user and a computer-based device by making a physical touch on the screen. As the name implies, gesture recognition aims to identify the physical motions, or "gestures," made by people. For example, one could instruct the gadget to start a specific app and carry out other operations by waving hand in a precise pattern in front of it. In this gesture recognition and control project, the computer camera interprets the movements of human hand. This information is subsequently used as input by the computer camera to handle various computer activities as well as applications. Gesture recognition is the foundation of gesture control technology. You may think of gesture recognition as a means for computers to start deciphering body language. It creates a richer connection between computers and people than the simple user interfaces, like the keyboard and mouse. Gesture control systems interpret and recognize human body gestures to enable user interaction with a computer system.

Gesture recognition is a topic of active research in Human Computer Interaction (HCI) technology. In the realm of Human Computer Interaction, gesture recognition is a well-liked and in-demand analytical technique. It has applications in a variety of jobs, including controlling robots, managing virtual environments, translating sign language for medical applications, automating homes, etc. The consumer electronics industry, the automobile industry, the healthcare industry, and the educational sector are some of the industries that have embraced gesture control technology. Recent HCI research has taken on a special significance. Due to its competence, the hand is the most useful communication tool in all the body parts. The term "gesture" is used in many situations involving human motion. Only certain human motions, particularly those of the hands, arms, and face which are referred to as gestures are instructive. The ability for machines to recognize and categorize hand gestures is provided by hand gesture identification, which is regarded as a vital component of Human Computer Interaction (HCI). With the development of current technology and the widespread use of hand movements or hand gestures in daily communication, the hand gesture identification is treated as a crucial part of HCI. Recently, the study of hand gestures has seen an increase in interest and popularity. Moreover, it permits a straightforward and natural method of interaction.

The goal of this project is to create an interface that dynamically captures human hand gestures and controls various functions of the computer device using hand movements. This will improve the entire experience of using a computer without the mouse or keyboard and using hand gestures to control the fundamental operations. As humans, we always look for ways to make our lives easier by the use of technology. Moreover, with the help of gesture control, even those with disabilities will be able to operate computers with ease.

II. LITERATURE SURVEY

In terms of popularity, gesture control technologies have eclipsed more traditional mechanical communication techniques. The domain market is divided into many categories depending on a variety of criteria, such as location, assistive robots, sign language recognition technology, immersive gaming, smart Televisions, virtual controllers, virtual mice, etc. Myron W. Krueger initially proposed the method of hand gesture-based computer

communication in 1970 [1]. This study [2] investigates the use of optical mouse sensors to create a motion based HCI device called a laser spot tracking system. The primary objective of the work is to characterize laser speckle detection using optical mouse sensors. As a precise, rapid, and compact sensing method based on the laser speckle displacement measurement capacity of optical mouse sensors, a low-cost optical mouse sensor is utilized to record the motion of the laser spot. The fundamental goal of gesture recognition is to recognize a specific human gesture and give information about it to the computer. The ultimate objective is to create computer systems that can identify human motions and remotely operate objects. Nowadays, most human-computer interaction (HCI) involves the use of keyboards or mouse, but the capacity to reliably recognize human gestures has increased the importance of computer vision-based techniques [3]. Paper [4] suggests hand gestures recognition for TV management. An actual HCI system based on gestures that uses a single monocular camera and reaches out to the HRI case has been built in this study [5]. A Convolution Neural Network classifier is used by the created system to comprehend its features and identify the gestures. A key tool for the real-time, dynamic gesture identification process is the Hidden Markov Model [6]. The HMM approach is practical and designed to function in static contexts. In order to create the best possible state sequence for pattern recognition, the approach uses the HMM's LRB topology along with the Baum Welch Algorithm for training, the Forward and Viterbi Algorithms for testing, and the Viterbi Algorithm for evaluating the input finding sequences. The system in this paper [7] looks to be simple to use in comparison to more recent systems or command-based systems, however it is less effective at spotting and recognizing patterns. Upgrading the system and make an effort to build a more robust algorithm for both detection and recognition despite a confusing background and typical lighting conditions. Also, the system must be upgraded to include several extra gesture categories because it was only designed to support six. For those with disabilities, a gesture control and hand recognition system was developed in this paper [8]. In this study, wearable hand gesture devices were employed with surface electromyography (sEMG) sensors, and an artificial neural network was primarily used as a classifier for hand gestures used in sign language. The dataset in this study [9] had an overfitting issue. The Python programming language, OpenCV libraries and Linux framework were used to construct the real-time, vision-based system. A virtual gesture control mouse method is created to assist the mouse pointer and enable real-time camera functionality for the mouse's functions [10]. In an effort to perform mouse indicator activities and icon selection, this technique is also constructed on the foundation of picture comparison and motion recognition mechanisms. This paper [11] focused on recognition systems that use vision. The skin tone color model, thresholding approach, template matching, and PCA were the variables that served as the foundation for the database of hand motions. The first phase is hand segmentation, which is accomplished by using the skin color model. The second step is the separation of the hand picture from the surrounding environment. A format-based matching is created in the final PCA stage in order to identify the gesture. Techniques for recognizing hand gestures using vision are examined in surveys spanning the previous 16 years. Further, this paper [12] analyses 26 publicly accessible hand gesture datasets in the vision-based hand recognition. In this paper [13], a vision-based hand gesture recognition application for HCI was built. A real-time system to stop mouse movements in windows without utilizing ANN training and by using a color-based detection technique is developed. According to [14], a system that primarily records human hand gestures onto a disc, converts them into binary images by extracting frames from each video one at a time and then forms 3D Euclidian space for binary images to classify hand motions has been proposed. Back propagation technique and supervised feedforward neural network are utilized in the training phase because they are suitable for simple gestures with limited background complexity. The approach for identifying the uncover hand with no color cap is suggested in this paper [15]. The Hue image is created by altering the RGB image. For localization and segmentation, the arm region removal approach and the hand skin segmentation technique are used. In this paper [16], a method is developed to identify hand motions in real-time photographs that are gathered. The hands that are present for categorization have a number of intersecting locations. This technique is also being used to address a few issues with gaming consoles like the Xbox, PS4, and smartphones. To address the issue, a clever solution is designed. The hand motions are recognized in the real-time photos using Python 3.9 and MediaPipe.

III. METHODOLOGY

One of the methods that can recognize a hand gesture in a real-time video in everyday life is the hand gesture recognition technology. The goal of this project is to design a hand gesture recognition and control system. One of the most crucial and essential concepts in computer vision is hand gesture control. The HCI systems that combine hand processing tasks like hand detection and hand gesture identification are now practical, all thanks to recent advancements in information and media technology.

The project encompasses various libraries such as OpenCV, Numpy, Pycaw, MediaPipe, PyAutoGUI. Computer vision and image processing activities can be performed with OpenCV. The features it offers include tracking, hand identification, and object detection. NumPy, a library for the Python programming language, supports large, multi-dimensional arrays and matrices as well as a large variety of high-level mathematical operations that may be carried out on these arrays. Pycaw (Python Core Audio Windows Library) is a volume-controlling Python Audio Control Library. Google's open-source MediaPipe machine learning library offers encapsulation for python and other languages and includes some solutions for gesture detection. A high-fidelity hand and finger tracking solution, MediaPipe Hands is also used in the project to recognize the motion and shape of hands which plays a key role in enhancing user experience across a range of technological platforms. It employs Machine Learning to extrapolate 21 crucial 3D hand details from a single frame. It is used to obtain the

ISSN No:-2456-2165

coordinate points of the fingertips of hand. The Python library PyAutoGUI enables users to simulate keyboard button pushes as well as mouse pointer clicks and movements. It has several features such as (1) moving the mouse and clicking in other applications' windows (2) sending keystrokes to other applications (for example, to fill out forms) (3) Capture screenshots, and when given an image (such as a button or checkbox), locate it on the screen (4) Find a window for an application, and move, resize, maximize, or close it (currently only available on Windows) and (5) Display alert and message boxes.

In this machine learning hand gesture control system project, Python and OpenCV are implemented to create a real-time hand gesture controller. The working principle of the project includes the camera which is used to detect the hand gestures with the help of associated points. The distance between the desired points on the hand is calculated and the associated gestures are recognized by the camera. The gestures are thereafter interpreted by the MediaPipe library, and the linked function is performed on the computer device. Various gestures are included in the project which perform tasks such as controlling the volume of device, taking screenshots, closing desktop applications, putting the device to sleep, system shut down, screen-lock, screen-unlock. Other actions related to controlling the Word Document like closing a file, saving a file, printing a file, etc. are also implemented in the project. When a word document is open and the save gesture is made, the document will be saved. Similarly, when the print gesture is made, the printer dialogue box will open. The gesture designated for close/exit will similarly close the active application. If no applications are running, the system will shut down automatically. It functions similarly to Windows PC's Alt+F4 key press.

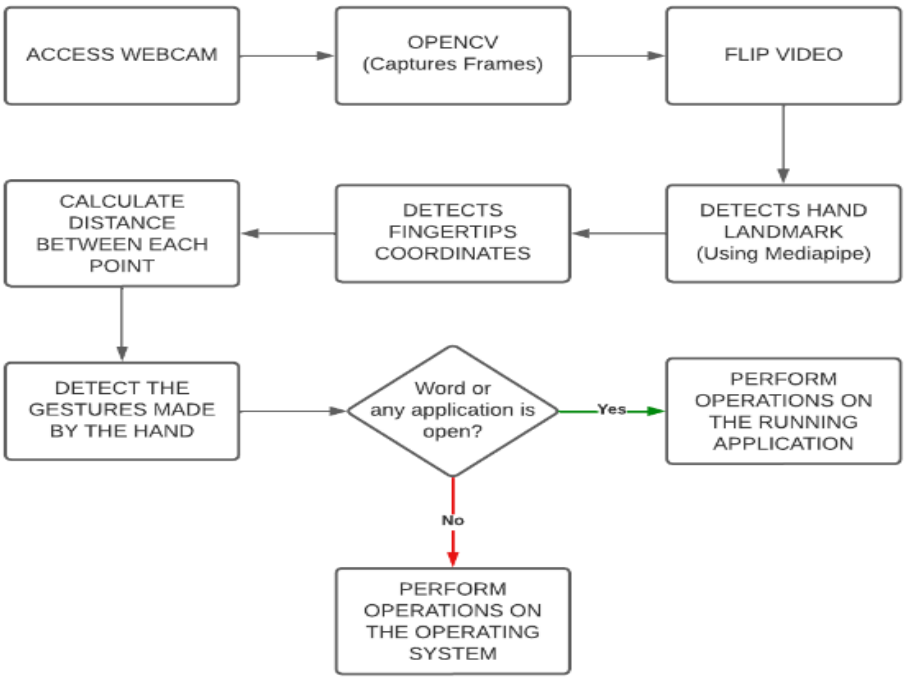
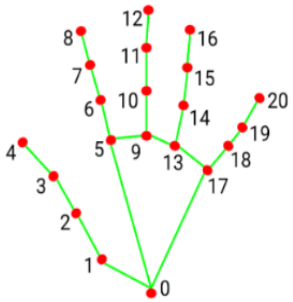


Fig. 1: Methodology Flowchart

The methodology of the project is explained stepwise in Fig 1. It is initiated when the hand movement is recognized through the camera. OpenCV captures the hand gestures by capturing the frames. The video accessed through the camera is flipped. The hand gesture language is detected using MediaPipe library and then the fingertip coordinates are detected. Further the distance between each point on the hand is calculated. The final gesture made by the hand is detected after calculating the distance between the points. If any application such as Word Document is open, then the associated gestures are performed on the running application. However, if no application is open, then the gestures are performed on the operating system itself.

Initially, the video input is obtained from the computer's primary camera and then the image is converted into RGB form, and the processing of the image is completed. The primary camera's video input is utilized to generate the video input for the MediaPipe Hand module, which is used to detect hands. The initialization and configuration of the discovered hands are finished by 'MpHands.Hands()'. Using 'mp.solutions.drawing_utils', the connections and landmarks are depicted on the detected hand. It uses a neural network to predict the locations of 21 key points on the hand, including the tip of each finger, the base of each finger, the center of the palm, and the wrist. The points 2 - 4 represent the thumb, points 5 - 8 represent the index finger, points 9 - 12 represent the middle finger, points 13 – 16 represent the ring finger and points 17 – 20 represent the little finger.



0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

Fig. 2: Hand Landmarks [17]

The list of hand components, or the number of points on the hand, as detected by the MediaPipe Hand module, is produced as an empty list. It also examines whether there are several hands on the input. Then the key points of the fingertip are specified in a lmList i.e. lmList[8][1], lmList[8][2]. In this way the x and y coordinates of a key point are created (such as: x, y = lmList[8][1], lmList[8][2]). After obtaining the coordinates for the key points a miniature opaque circle is drawn at the coordinates using cv2.circle() function. These circles are then connected by drawing solid lines using c2.line() function.

Further, the distance between the coordinates is calculated using the hypot() function imported from the math library. The distance is calculated to differentiate between different gestures and to identify the gestures distinctively. Then it is detected whether an application is in the running state or not. If an application is running, for example, Word Document is open, then a set of gestures can be used to perform different operations on the word file such as print, save, close, etc. If no application is running then the same set of gestures can be used to perform operations such as volume up and down, system shutdown, screenshot, etc.

As different gestures made by the hand have different distances between the coordinates, using these different distances we can differentiate between the different gestures. For example, if a fist is made then the distance between the finger tips become very close to zero so we define a range like 0-15 for the distance between the finger tips, but if a 'yo gesture' is made then the distance between the fingertip of thumb, middle and ring finger becomes close to zero and the distance between little and index finger becomes very large, thereby giving it a higher range. Once the gesture is identified, the pycaw library for volume controls and pyAutoGUI library is used to perform other different operations on the operating system or the current running application.

Details of Hardware & Software for the implementation of the project are as follows:

A. Hardware requirements: -

- Pentium i3 Processor System
- 4 GB Ram
- 500 GB Hard Disk Space
- LED Monitor
- Input Devices like Keyboard, Camera, Mouse

B. Software requirements: -

- Python 3.8 Programming Language
- Windows 10 Operating system

IV. RESULTS AND DISCUSSIONS

After the text edit has been completed, the paper is ready for the template. Duplicate the template file by using the Save As command and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all of the contents and import your prepared text file. You are now ready to style your paper; use the scroll down window on the left of the MS Word Formatting toolbar.

The project has resulted in identifying the hand gestures appropriately and performing the associated activities such as volume control, screenshot, closing a file, save and print operations on a file, device sleep and shut down successfully. The implementation of the hand gesture control system along with the screenshots can be seen below. Actions associated with specific gestures were captured by the device camera and the operations for various tasks were performed.

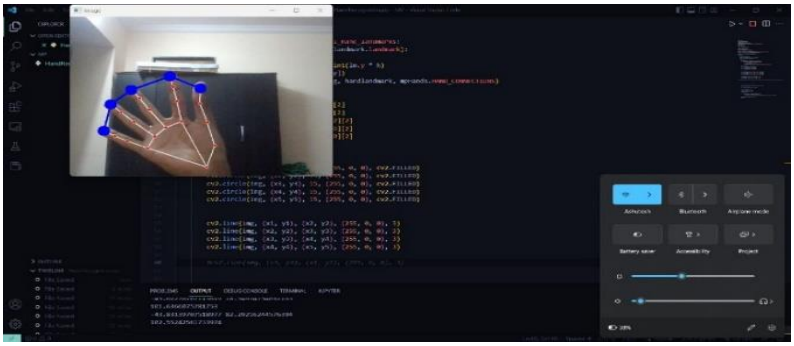


Fig. 3: Volume Control Hand Gesture

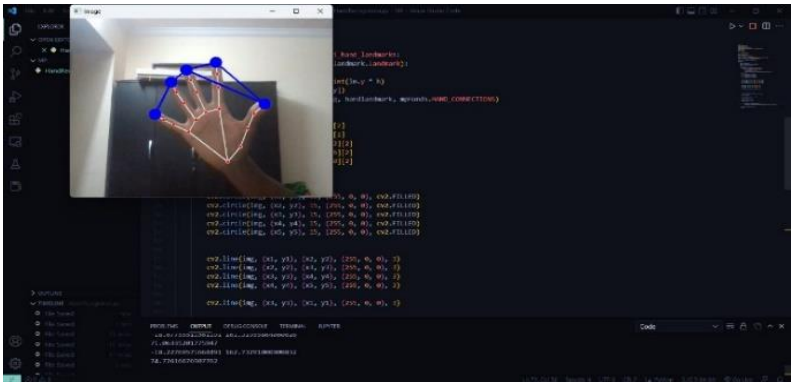


Fig. 4: Screenshot Hand Gesture

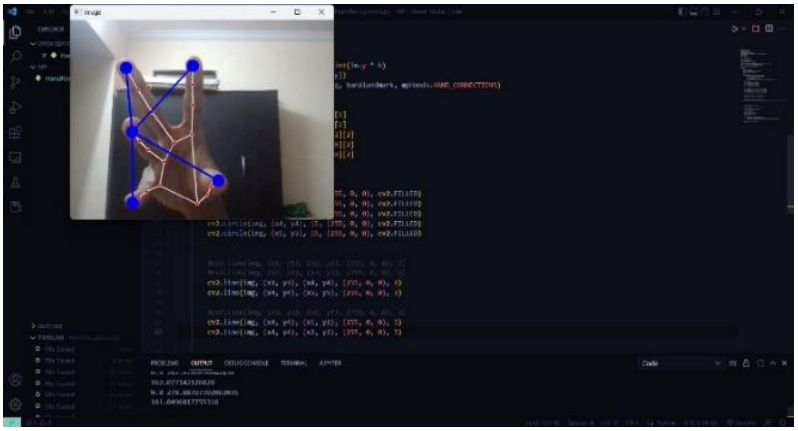


Fig. 5: Hand Recognition

Gesture control technology has advanced greatly since the 1980s, when the first research was conducted. Based on the project's findings, it can be concluded that hand gesture identification and control using Python and OpenCV can be developed utilizing the theories of hand segmentation and the hand detection system, which apply the Haar-cascade classifier. To sum it up, this system has achieved project goals such as (1) To establish a comprehensive system for hand gesture detection, interpretation, and recognition using Python and OpenCV and (2) To create the numbers and sign languages of hand gestures displayed in the system to fulfil the project's title.

The implementation of additional gestures can be included in the system's future recommendations so that users with varying skin tones and hand sizes will be able to effortlessly complete more tasks. The right hand with a specified area in ROI is the sole hand used in the current system to make gestures. As a result, the technique may be improved to allow for the employment of both user hands to execute various signs in conjunction with computer processes. A recognition system for both the hands working simultaneously can be implemented to level the project on a higher scale. This can further be used in gaming technologies, thus enhancing the user experience. Algorithms for background subtraction can also be employed for better detection and tracking of the hand thereby resulting in better performance. In the future, when users will be able to interpret a gesture's meaning into a sign or number and vice versa, the Graphical User Interface (GUI) can be implemented. Nevertheless, technology is constantly evolving, so implementing a more advanced version of the same technology might undoubtedly be advantageous and will result in a better interaction of human and computer devices by elevating the Touch User Interface (TUI).

ACKNOWLEDGMENT

We would like to extend our sincere gratitude to all the people who played a crucial role in the completion of our project. We would like to express our deepest gratitude to our coordinator, Prof. Arathi Boyanapalli, for her continuous support throughout the duration of the project. We would also like to express our sincere gratitude to our colleagues, whose efforts were instrumental in the completion of the project. Through the exchange of interesting ideas and thoughts, we were able to present a project with correct information. We also want to express our gratitude to our parents for their personal support and encouragement to us to pursue our own paths. In addition, we would like to express our sincere gratitude to our Director, Dr. Atul Kemkar, and our Head of Department, Dr. Aparna Bannore. We also thank our other faculty members for their invaluable contribution in providing us with the required resources and references for the project.

DECLARATION

We confirm that the contents of this written submission reflect our own ideas and opinions. Where we have incorporated ideas or words from other sources, we have provided appropriate citations and sources to confirm the original source. We also confirm our commitment to academic integrity and honesty. We have not

misrepresented, fabricated, or falsified any idea, data, fact, or source in our submission. As a result, we maintain the authenticity and honesty of our work. We pledge to maintain the highest level of academic ethics in all our academic endeavors.

REFERENCES

[1]. M. Kruger, Artificial reality II, Addison-Wesley Reading (Ma), 1991.
[2]. Guoli Wang, (2010). Optical Mouse Sensor-Based Laser Spot Tracking for HCI Input, proceedings of the 2015 Chinese Intelligent Systems Conference: Volume 2, pp.329-340.
[3]. J. C. Manresa, R. Varona, R. Mas and F. Perales, "Hand tracking and gesture recognition for human computer interaction", Electronic letters on computer vision and image analysis, vol. 5, pp. 96-104, 2005.
[4]. W. Freeman and C. Weissman, "Television control by hand gestures", Proc. of Intl. Workshop on Automatic Face and Gesture Recognition 1995, pp. 179-183, 1995.
[5]. Pei Xu, "A Real-time Hand Gesture Recognition and Human-Computer Interaction System", arXiv:1704.07296v1 [cs.CV] 24 Apr 2017.
[6]. J.R Pansare, Malvika Bansal, Shivin Saxena, Devendra Desale, "Gestuelle: A System to Recognize Dynamic Hand Gestures using Hidden Markov Model to Control Windows Applications", International Journal of Computer Applications (0975 – 8887) Volume 62– No.17, January 2013.
[7]. Ram Pratap Sharma and Gyanendra K. Verma, "Human Computer Interaction using Hand Gesture", Eleventh International Multi-Conference on Information Processing-2015 (IMCIP-2015).
[8]. Varun, Kollipara Sai, I. Puneeth, and T. Prem Jacob. "Hand gesture recognition and implementation for disables using CNN'S." 2019 International Conference on Communication and Signal Processing (ICCSP). IEEE, 2019.
[9]. Yasen, Mais, and Shaidah Jusoh. "A systematic review on hand gesture recognition techniques, challenges and applications." PeerJ Computer Science 5 (2019): e218Anna De Liddo, Ágnes Sándor, et.al, (2012). Contested Collective Intelligence: Rationale, Technologies, and a Human-Machine Annotation. Computer Supported Cooperative Work (CSCW) Volume 21, Issue 4–5, pp 417–448.
[10]. Shibly, Kabid Hassan, et al. "Design and development of hand gesture based virtual mouse." 2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT). IEEE, 2019.
[11]. Ahuja, Mandeep Kaur, and Amardeep Singh. "Static vision based Hand Gesture recognition using principal component analysis." In MOOCs, Innovation and Technology in Education (MITE), 2015 IEEE 3rd International Conference on, pp. 402-406. IEEE, 2015.
[12]. Pisharady, Pramod Kumar, and Martin Saerbeck. "Recent methods and databases in vision-based hand gesture recognition: A review." Computer Vision and Image Understanding 141 (2015): 152-165.
[13]. Chetan Dhule, Trupti Nagrare, "Computer Vision Based HumanComputer Interaction Using Color Detection Techniques." Fourth International Conference on Communication Systems and Network Technologies, pp.934-938, 2014.
[14]. Murthy and Jadon, "Hand Gesture Recognition using Neural Networks," 2nd IEEE International Advance Computing Conference (IACC), pp. 134–138, (2010).
[15]. Xu, Yishen, Jihua Gu, Zhi Tao, and Di Wu. "Bare hand gesture recognition with a single color camera." In Image and Signal Processing, 2009. CISP'09. 2nd International Congress on, pp. 1-4. IEEE, 2009.
[16]. Lavanya Vaishnavi D. A., Anil Kumar C., Harish S., Divya M. L. "MediaPipe to Recognise the Hand Gestures", WSEAS TRANSACTIONS on SIGNAL PROCESSING DOI: 10.37394/232014.2022.18.19Lavanya, July 2022.
[17]. Hands | mediapipe (google.github.io) | https://google.github.io/mediapipe/solutions/hands.html