CONTROLLING MOUSE AND OTHER OPERATIONS USING HAND GESTURES

Sri. K. N. V. Suresh Varma

Asst. Professor, Department of Electronics and Communication Engineering

Sagi Ramakrishnam Raju Engineering College (JNTUK Affiliated)

Bhimavaram, Andhra Pradesh

Email: knvsureshvarma@yahoo.co.in

Sambhav Surana, R. Pranay Krishna, SK. M. Jakeer Hussain, M. Gnaneswar

4th B.Tech, Department of Electronics and Communication Engineering

Sagi Ramakrishnam Raju Engineering College (JNTUK Affiliated)

Bhimavaram, Andhra Pradesh

Abstract

This paper demonstrates an approach for controlling mouse movements and performing other keyboard operations using hand gestures given by the user. Some of the available approaches simplify such operations by adding more buttons to the already bulky hardware, but our approach is to eliminate the usage of external hardware. We have planned to use Computer Vision technology to achieve this. Computer vision technology would use a camera which is already inbuilt in most devices these days. Through Computer Vision technology, we will acquire the image, divide it into segments and recognize the gesture. Our proposed system performs mouse operations such as mouse movement, left-click, right-click, drag and drop, double-click, scroll, and other operations like volume control, brightness control, zoom in and out, tab switch and hide and show desktop, thus eliminating the need for external hardware. This proposed software is built using Python. This proposed model worked well with an accuracy of 99.07% which is best when compared to previous existing models.

Keywords: Image Processing, Python, Computer vision, OpenCV, Mediapipe, pynput, Colour detection, Skin detection, data gloves, Virtual mouse, Hand tracking, Palm detection, Deep learning.

1. Introduction

In computer systems, the most used process or technique for controlling the mouse cursor is through the usage of a mouse or touchpad. Even though it is comfortable to use, it is not hardware free. With the advancement of technology, many devices are being designed to be hands-free. Technologies like speech recognition are being used in order to minimize the physical interaction between a human and an electronic device. We are trying to contribute to this domain by designing a software that makes use of machine learning and image processing technologies. Our proposed model helps in achieving the things done by the mouse without using the hardware. We have designed this system in a way that replaces the external hardware with a hand gesture control-based system.

Hand gesture control-based system works on computer vision technology which captures a set of images in real-time. These captured sets of images are given as input to our model. The whole system works using Image Processing. The system consists of image acquisition, image segmentation, gesture recognition, and event execution. This project mainly uses the OpenCV [1] and Mediapipe [2] libraries in python. Some other libraries are also used to control the mouse events and other operations.

The first step of our proposed model is to obtain the input images. The set of images is taken as input from the real world using the python library OpenCV [1]. Even though there are a lot of libraries to get the images, we used the OpenCV library as it is popular and supported on every operating system from Android to Windows. We had to set up the OpenCV library in our PyCharm IDE.

Importing the OpenCV library is a simple task. One can refer to the web to know how to import the library. We can use the high-end cameras (external) or webcam (inbuilt) for our model.

The second step is to detect the hand. There are some methods in previously proposed models where the set of images is converted into HSV [3]. After detecting the skin in hand, the contours are found using OpenCV processing. But we used another method using Mediapipe. In our proposed model, we used pre-trained datasets which are imported from Mediapipe [2]. Hand detecting and tracking [4] can be done using this Mediapipe library.

For the purpose of controlling and triggering mouse events and other operations, we used pynput [5], and keyboard libraries from python. In the end, our proposed model is converted into an application, i.e., .py to .exe.

2. Literature survey

There are some previously proposed models of virtual mouse control using hand gestures but there are some complications in their proposed models. One of the proposed models is by wearing a glove and using colour caps on fingers for detection of fingers. Also, gesture recognition is not accurate and mismatches events because of wearing gloves. In some other models, there is a failure in the detection of fingertips.

Quam in 1990, proposed the first hardware-based model to recognize the gestures [6]. In that model, the data gloves are used to recognize the gestures. The used data gloves are bulky and make it difficult for users to perform the operations.

Siam et al. in 2016 have applied the method of using colour caps [7] [8] to fingers for the detection of fingers. As a result, the use of markers made computational time low, and accuracy was increased. But this system's accuracy is decreased when the background is noisy. Also, the gestures used and events are fewer.

Manav, Madhur, Neha, and Radha 2021 proposed "Hand Gesture Recognition Based Virtual Mouse Events" [9] where the model works on skin detection. Because of skin detection [10], the library would detect multiple hands from the background which leads to complications in gesture recognition. This model also contains only fewer mouse operations.

Shriram, Nagaraj, Jaya, Shankar, Ajay in 2021 proposed "Deep Learning-Based Real-Time AI Virtual Mouse System Using Computer Vision to Avoid COVID-19 Spread" [11] where the model works using Mediapipe [12] and machine learning algorithm [10]. But the proposed paper contains only fewer mouse operations and no other operations such as volume, brightness control, zooming etc.

3. Methodology

The motive behind this project is to perform the following operations:

Cursor movement, left click, right click, double click, scroll, drag & drop, volume control, brightness control, zoom in, zoom out, tab switch, and hide & show desktop.

This is mainly done in 5 phases:

1. **Image Acquisition**: The application requests camera access (external or inbuilt) and the live video feed is captured using the OpenCV [1] module. Every single image frame in the captured video is then processed. The video is continuously captured and processed till the program is terminated.
2. **Palm Detection:** The video frames are first converted from BGR to RGB [3] so that they can be further processed. Then the palm of the hand is detected using machine learning algorithms. In this project, the Mediapipe module is used for this purpose. Mediapipe is an ML pipelining solution

developed by Google, which solves various machine learning challenges. The ML algorithm used by mediapipe [2] to detect the palm is KNN algorithm.

3. **Hand Landmark Identification**: Through mediapipe, we identify 32 different landmarks on the palm. These landmarks are later used to identify the gesture.
4. **Gesture Recognition**: Various gestures have been designed based on the hand landmarks and each gesture triggers a unique operation. A gesture is a combination of different fingers lifted together.
5. **Operation**: After the gesture has been recognized, the corresponding operation is performed. Modules like pynput, screen_brightness_control, keyboard, mouse, wx, and pycaw are used to access and control the hardware of the system being used.

Basic Flow of the application

Step 1: Start the application.

Step 2: Video is captured using OpenCV

Step 3: A video frame is processed, and hand is detected using hand tracking

Step 4: Hand Landmarks are identified.

Step 5: Gestures are recognised.

Step 6: An event occurs based on the gesture used.

Step 7: The process is continued until the application is ended.

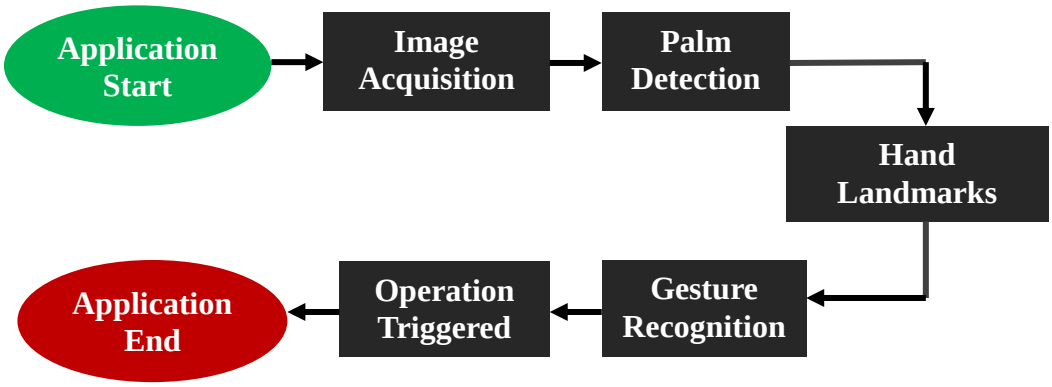

Figure 1: Basic Block Diagram

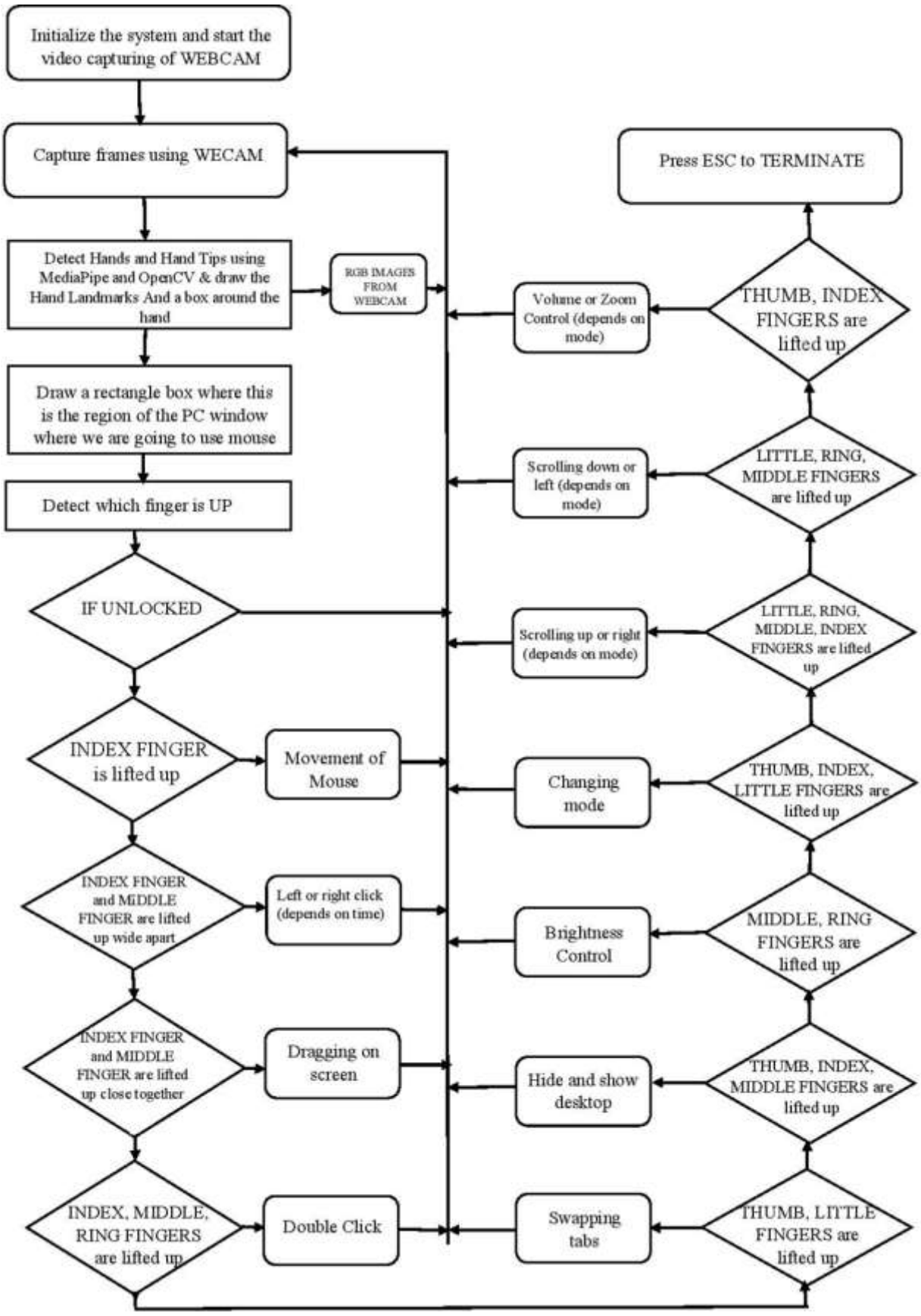

Figure 2: Flowchart of the project

Gestures

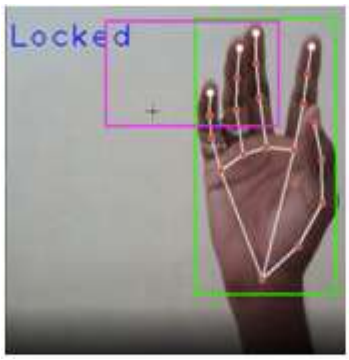

Figure 3: Hand Tracking

Figure 4: Lock / Unlock

Figure 5: Cursor

Figure 6: Left Click

Figure 7: Right Click

Figure 8: Drag & Drop

Figure 9: Double Click

Figure 10: Hide & Show

Figure 11: Switch Tabs

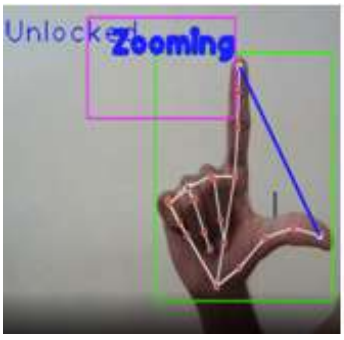

Figure 12: Zoom In & Out

Figure 13: Scrolling

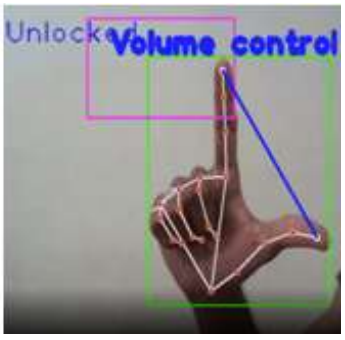

Figure 14: Volume

4. Result and Evaluation

In our proposed model, the concept of controlling mouse and performing other operations is given.

Table 1: Accuracy of each operation

FINGER LIFTED	Operation Performed		Success	Failure	Accuracy %
INDEX, MIDDLE, LITTLE	Lock / Unlock		100	0	100
THUMB, INDEX, LITTLE	Switching Mode		98	2	98
	Mode 1	**Mode 2**			
INDEX	Movement	Movement	98	2	98
INDEX, MIDDLE	Left click	Left click	100	0	100
INDEX, MIDDLE	Right click	Right click	100	0	100
INDEX, MIDDLE	Dragging	Dragging	99	1	99
INDEX, MIDDLE, RING	Double click	Double click	98	2	98
INDEX, MIDDLE, RING, LITTLE	Scrolling Up	Scrolling Right	99	1	99
MIDDLE, RING, LITTLE	Scrolling Down	Scrolling Left	97	3	97
THUMB, LITTLE	Swapping tabs	Swapping Tabs	100	0	100
THUMB, INDEX, MIDDLE	Hide/Show desktop	Hide/Show desktop	100	0	100
THUMB, INDEX	Volume control	Zoom in/out	100	0	100
MIDDLE, RING	Brightness control	Brightness control	99	1	99
Result			1285	15	**99.07 %**

An experimental test has been conducted to calculate the results. The test was conducted 20 times by 5 persons. Thus, there are total of 1300 gestures. This test is conducted in every condition with every distance. Each person tested this model 5 times in good light condition, 5 times in average light condition, 5 times at close distance to webcam, and 5 times at long distance from webcam.

From the accuracy we obtained from the results, we can finalise that this model performed well.

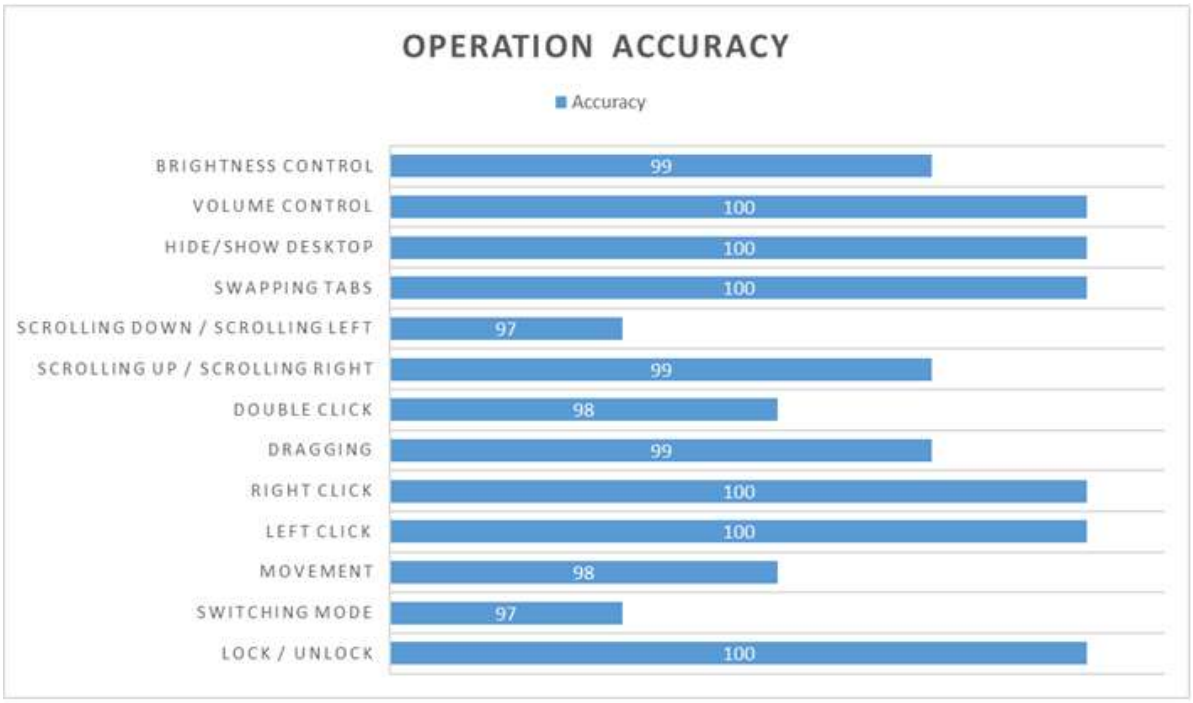

Figure 15: Bar graph of Accuracy of each operation

Table 2: Comparison with existing models

Existing Models	Accuracy (%)
Virtual Mouse system using RGB-D images and fingertip detection [13]	96.13
Recognition based on palm and finger [14]	78
Deep learning based Virtual mouse system [11]	99
Hand Recognition based virtual mouse system [9]	98.47
Our proposed system	99.07

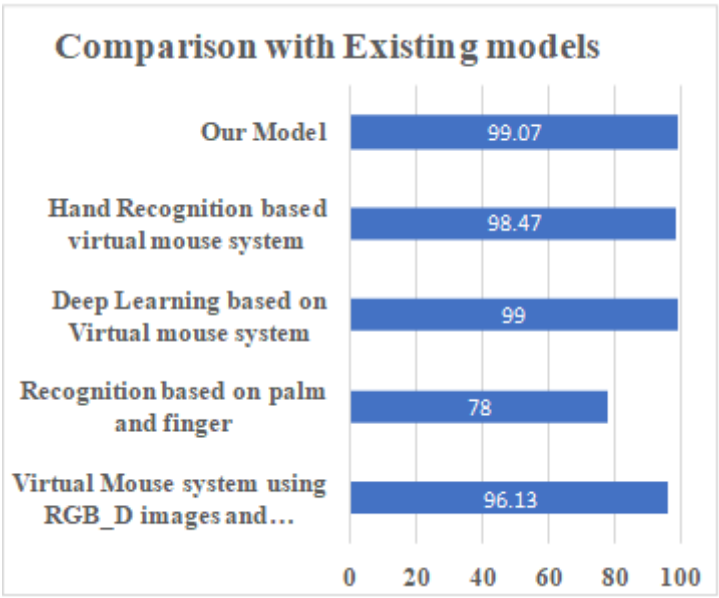

Figure 16: Bar graph of comparison with existing models

Shown below are a few visible outputs for their corresponding gesture input.

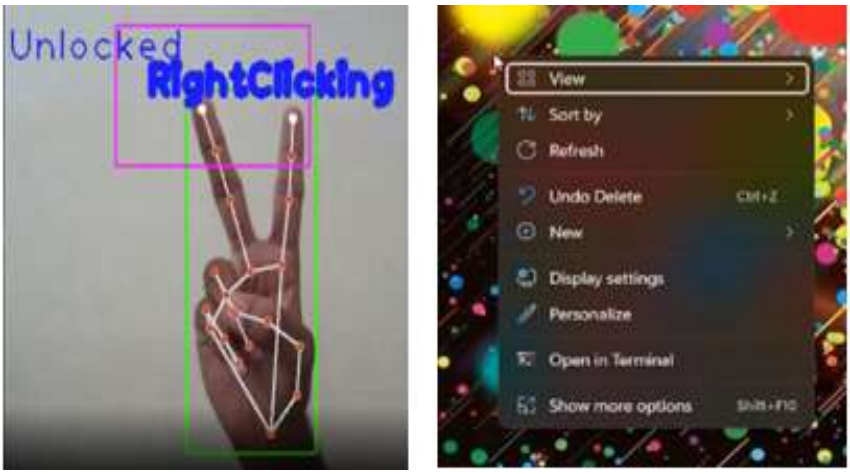

Figure 22: Right Click Output

Figure 23: Volume control output

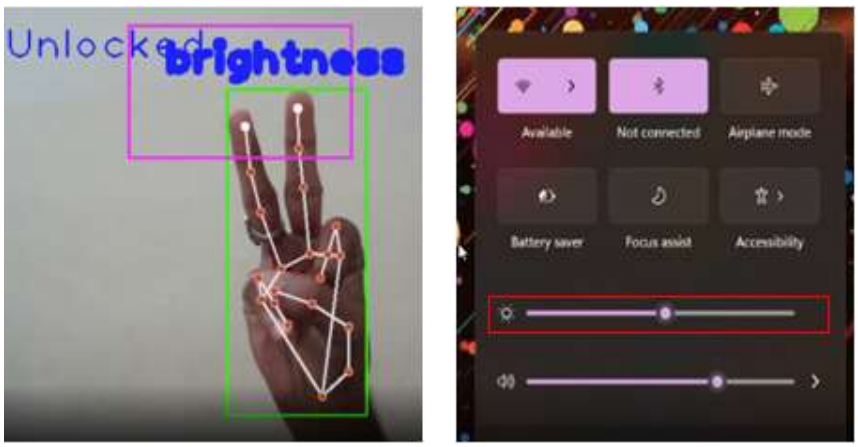

Figure 24: Brightness Control Output

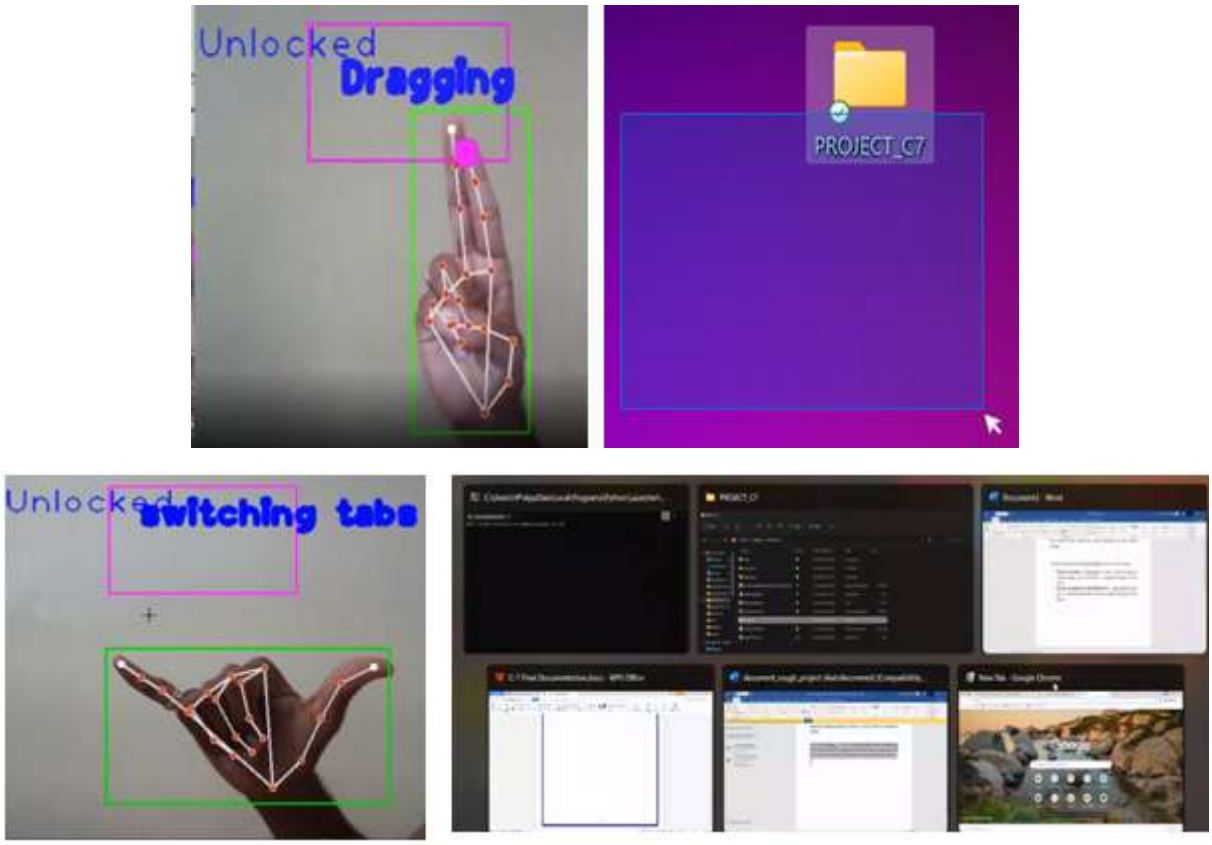

Figure 25: Tab Switch Output

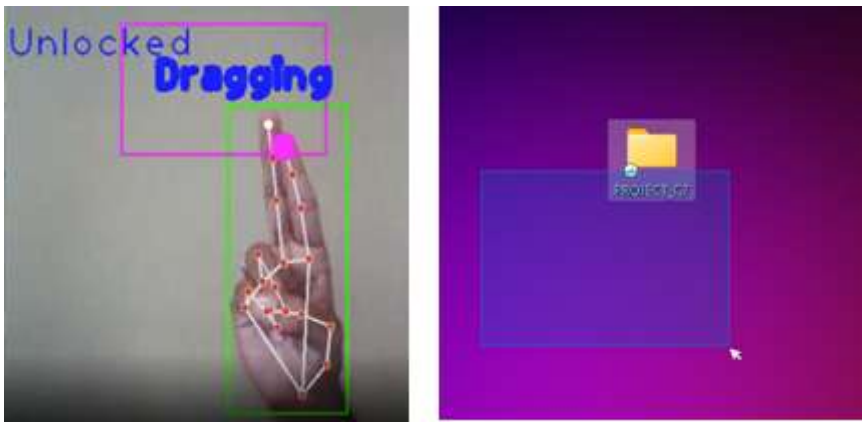

Figure 26: Dragging Output

5. Conclusion

The main aim of our proposed model is to control the physical mouse and few other operations. Our proposed system successfully achieved the expected output for respective gestures. This proposed system achieved by the webcam or inbuilt camera which captures the hand and detects the palm and fingers. By the hand gesture captured, system performs the operation. From the results, we can conclude that the model performed well with greater accuracy and our model also overcame the limitations of previously proposed system. Also, our system has better accuracy than previous models. Since, this proposed model has better accuracy it could be used in real-time applications. Also, as we eliminated the use of external hardware, the scope of COVID 19 spread will be decreased. This model has some limitation such as, there should be sufficient light on the user

so that the system can take set of images with clean clarity of palm. This limitation can be overcome by running this model in sufficient light.

References

[1] "OpenCV," [Online]. Available: https://opencv.org/ .
[2] "Mediapipe," [Online]. Available: https://mediapipe.dev/ .
[3] "OpenCV operations," [Online]. Available: https://www.javatpoint.com/opencv-basic-operation-on-images .
[4] "Hand Tracking," [Online]. Available: https://ai.googleblog.com/2019/08/on-device-real-time-hand-tracking-with.html .
[5] "pynput," [Online]. Available: https://pypi.org/project/pynput/ .
[6] D. Quam, "Gesture recognition with a DataGlove," IEEE Conference on Aerospace and Electronics, pp. 755-760, 1990.
[7] J. A. S. M. h. K. Sayem Mohammad Siam, "Human Computer Interaction Using Marker Based Hand Gesture Recognition," 2016.
[8] T. G. V. K. a. S. M. V.V.Reddy, "Virtual Mouse Control Using Colored Finger Tips and Hand Gesture Recognition," 2020 IEEE-HYDCON, pp. 1-5, 2020.
[9] M. R. N. L. a. R. S. M. Ranawat, "Hand Gesture Recognition Based virtual Mouse Events," 2021 2nd International Conference for Emerging Technology(INCET), pp. 1-4, 2021.
[10] R. Puri, "Gesture Recognition Based mouse Events," International Journal of Computer Science and Information Technology, vol. 5, 2014.
[11] B. N. J. J. S. S. P. A. S. Shriram, "Deep Learning-Based Real-Time AI Virtual Mouse System Using Computer Vision to Avoid COVID-19 Spread," Journal of Healthcare Engineering, p. 8, 2021.
[12] A. K. T. a. K. K. S. B. J. boruah, "Development of a Learning-aid tool using Hand Gesture Based Human Computer Interaction System," 2021 Advanced Communication Technologies and Signal Processing (ACTS), pp. 1-5, 2021.
[13] N.-H. H. H. -J. Y. S. -H. K. a. G. S. L. D.-S. Tran, "Real-time virtual mouse system using RGB-D images and fingertip detection," Multimedia Tools and Application-Multimedia Tools and Applocations, vol. 80, p. 7, 2021.
[14] A. S. N. A. S. P. a. J. S. N. A. Haria, "Hand Gesture recognition for human computer interaction," Procedia Computer Science, vol. 115, pp. 367-374, 2017.

