A Vision Base Application For Virtual Mouse Interface Using Hand Gesture

Sankha Sarkar
Department of Computer Science and Engineering
JIS College of Engineering
Kalyani, West Bengal, India

Indrani Naskar
Department of Computer Science and Engineering
JIS College of Engineering
Kalyani, West Bengal, India

Sourav Sahoo
Department of Computer Science and Engineering
JIS College of Engineering
Kalyani, West Bengal, India

Sayan Ghosh
Department of Computer Science and Engineering
JIS College of Engineering
Kalyani, West Bengal, India

Sumanta Chatterjee
Department of Computer Science and Engineering
JIS College of Engineering
Kalyani, West Bengal, India

Abstract– This paper proposes a way of controlling the position of the mouse cursor with the help of a fingertip without using any electronic device. We can be performing the operations like clicking and dragging objects easily with help of different hand gestures. The proposed system will require only a webcam as an input device. Python and OpenCV will be required to implement this software. The output of the webcam will be displayed on the system's screen so that it can be further calibrated by the user. The python dependencies that will be used for implementing this system are NumPy, Autopy and Mediapipe.

Keywords :- *OpenCv,Autopy,Mediapipe Numpy; Calibrated, Gesture.*

I. INTRODUCTION

With the developing technologies within the twenty-one century, the areas of virtual reality devices that we are using in our daily lifestyle, these devices are become more compact with wireless technologies like Bluetooth. This paper shows an AI virtual mouse system that take input of hand gestures and detection for fingertip movement to perform mouse operations in computer by using computer vision. The most important objective of the proposed system is to perform mouse operation like single click, double click, drag and scroll by employing an in-built camera or a web-camera within the computer rather than using a traditional mouse. Hand gesture and fingertip detection by using computer vision is employed as an individual's and Computer Interaction, simply referred to as HCI. Once we are using the AI virtual mouse system, we will track hand gestures and fingertips by using an in-built camera or web camera and perform the mouse cursor operations like scrolling, single click, double click. While employing a wireless or a Bluetooth mouse, that's not purely wireless some devices like the mouse, little dongle we've to attach to the PC, and also a battery cell to power the mouse. But in this system, the user uses his/her hand to manage the pc mouse operations. During this time, the camera captures images and store temporary and processes the frames and then recognizes the predefined hand gesture operations. Python programing language is mostly used for developing this kind virtual mouse system, and OpenCV library is additionally used for contract computer vision. The system makes use of the well-known python libraries like MediaPipe and Autopy for the tracking of the hands and fingertip in real-time and also, Pyput, and PyAutoGUI packages were used for tracking the finger traveling in the screen for performing operations like left-click, right-click, drag and scrolling functions. The output results and accuracy level of the proposed model showed very high, and also the proposed model can work in the dark envirment as well as 60cm far from the camera in real-world applications with the employment of a CPU without the employment of a GPU.

II. ALGORITHM USED FOR HAND TRACKING

A. Mediapipe

It is a cross-platform framework that's mostly used for building multimodal pipeline in machine learning, and it's an open-source framework by Google. MediaPipe framework is most useful for cross-platform development work since this framework is constructed by statistic data. The MediaPipe framework works as multimodal, which implies this framework is applied to face detection, facemesh, iris scanner pose detection, handdetection, hair segmentation object detection, motion tracking and objection. The MediaPipe framework is that the best option to the developer for building, analyzing and designing the systems performance in the form of graphs, and it's also been used for developing various application and systems within the cross-platform (Android, IOS, web, edges devices). The involving steps in our proposed system uses MediaPipe framework as pipeline structure configuration. This pipeline structure create and run in various platforms which allowing scalability in mobile and desktop system also. The MediaPipe package gives us three reliability, they're like performance, evaluation, and creation which isretrievingby the sensor data, and using a set of components. Mediapipe processing inside a graph which is defines flowing of packet between nodes.

Apipelinesstructures could be a graph that consists of components called calculators, where each calculator is connected by specific streams during which the information packets flow. Developers also are ready to replace or define custom calculators anywhere within the graph for creating their custom applications. The calculators and streams are combined to form a data-flow diagram. The graph (Figure 1) is made by using the MediaPipe module. Each calculator modules runs according to the timestamp become available.it used real-time graph system to define when the timestamp become available. A model of hand landmark which is consists of 21 locating points, as shown in Figure 1.

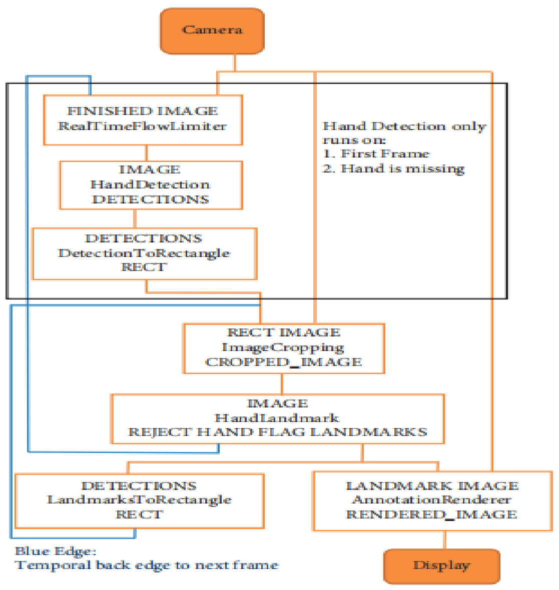

Fig. 1: MediaPipe hand recognition flowchart

B. Autopy

It is cross platform GUI automation module for python. That module keeps tracks of finger in this proposed system.Autopy track the fingertip and tell us which finger is up and which one is down .This process is happing by giving system an input in the form of 0 and 1.From this module the mediapipe module takes output and done the process and give the proper output. From this output opencv visualize everything and create the proper frame from image.

C. OpenCv

Open source computer vision library is a programming functions written in C++ mainly aimed at computer vision. It was licensed by Apache and introduced by Intel. This library is cross platform and free to use.it provides a features of Gpu acceleration for real time operation.opencv are used in wide areas like as 2D & 3D feature toolkits, Facial and gesture recognition system, mobile robotics.

III. METHODOLOGY

A. Importing the necessary library:

At very first install& import all the necessary library in preferred IDE.We are supposed to install onencv, numpy,mediapipe and autopy framework.After successfully install all the library in pycharm, our next is to import all that library into the code section so that we can make use this library in our system.

B. Initializing the capturing device:

Our next task istoinitialize the connected image capture device. We are make use of system primary camera as cap = cv2.VideoCapture(0).if any secondary image capturing device connected that is not used by this program.

C. Capturing the frames and processing:

The proposed AI virtual mouse system uses primary camera for capturing the frames. The video frames are processed and converted BGR to RGB format to find the hand in video frame. This color conversion is done by the following code:

```
check. img = cap.read()
imgRGB = cv2.cvtcolor(img. cv2.COLOR_BGR2RGB)
lmlist = handLandmarks(imgRGB)
if len(lmlist) !=0:
   finger = fingers(lmlist)
   x1, y1 = lmlist[0][1:]
   x2, y2 = lmlist[12][1:]
   x3 = numpy.interp(x1, (75, 640 - 75), (0, wScr))
   y3 = numpy.interp(y1, (75, 480 - 75), (0, hScr))
```

D. Initializing mediapipe and capture window:

After that we are Initializing mediapipe which is responsible for tracking the hand gesture and other actions. Then we are setting the minimum confidence for hand detection and minimum confidence for hand tracking by mainHand=initHand.Hands(min_detection_confidence=0.8, min_tracking_confidence=0.8).

That means it gives a success rate of 80% according to the performed action by the main hand. After that mediapipe draw the axis line across the palm and middle of fingers. That keeps track about finger movement and create connection with neighbor finger. Then we define the axis of the hand in the 2D format by defining the x-axis and y-axis value. By the help of this axis we are keep track about the finger that which one is up and which one is down. We also keep a record of the previous frame of x-axis and y-axis values in the form of 0 and 1.

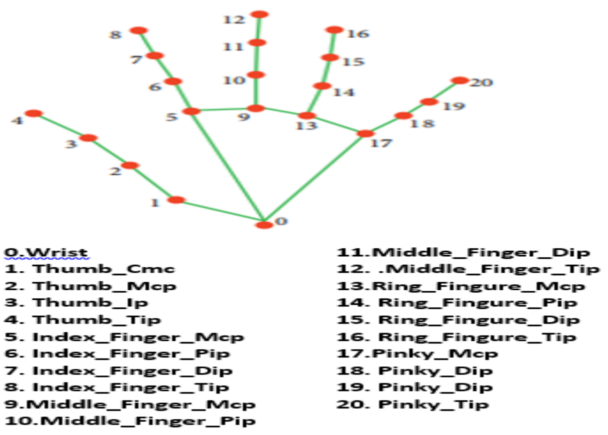

Fig. 2 : Co-ordinates or landmarks in the hand using Mediapipe

E. Defining hand landmarks:

After confirming the hand axis our next task is to define the full hand land mark capture. We are capture the hand land mark by the predefined mediapipe 21 locating point (Figure 2).These 21 joint or locating point are to be tracked by the mediapipe.If there is no landmark found the we print the default value that is empty. That's means no action need to be performed. Otherwise it returns the coordinate value of X,Y and Z axis.The thumb finger is not included in the hand landmark region. We are creating a loop in between 21 point and return the tip point value by drawing graph line in between finger. When we fold the finger or down the finger the lines are also down and the corresponding graph line are down. The gap n between the finger is not important in this point .The gap in between finger can become 5cm. Otherwise it get back to the initial position of finding. After locating height, width and center of the hand landmark we converts the decimal coordinate relative to the index according to the each image. After capturing the total hand landmarks a green rectangle box appeared around the hand where we can perform the mouse operation.

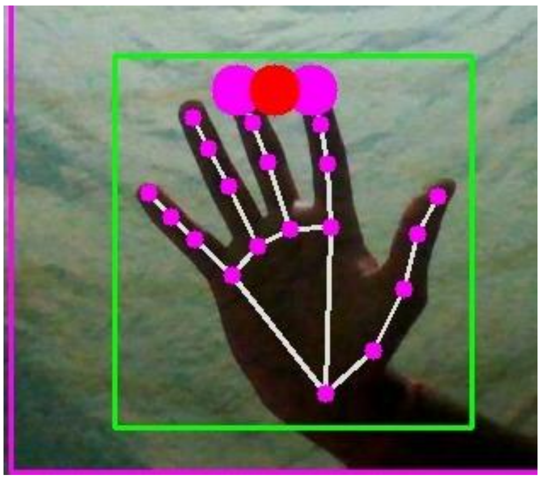
Fig3: capturing the hand landmarks

F. Defining the finger landmarks:

After initializing the axis point are defining the hand landmark. That is occupied the green dialog box and the finger height width. The height and width are measured by the captured image frame and counting the finger up down number by their movement. We are make use only the three finger. The thumb finger is not included in the hand landmark region. We are creating a loop in between 21 point and return the tip point value by drawing graph line in between finger. Otherwise it get back to the initial position of finding. After locating height, width and center of the hand landmark we converts the decimal coordinate relative to the index according to the each image. Here we are creating an array of size 4 which is storing the four index value of the different finger. The Tip _ID only considered the 4, 8,12,16,20 no index point. Then we are checking for the two condition (1) If thumb is up and (2) If finger are up except the thumb. In condition (1) we are checking **if** landmarks[tipIds[0]][1] > lmList[tipIds[0] - 1][1]:

Then append the fingerTip and returned to the finger landmark section. If condition (2) is satisfied then the program checked for which finger is up by their corresponding index ID's

```
. for id in range(1, 5):
if landmarks[tipIds[id]][2] < landmarks[tipIds[id] - 3][2]:
fingerTips.append(1)
else:
    finger Tips.append(0)
```

Otherwise it returned to the Fingertip function.

Color convert: Hand contour are the curve of the hand segmented image extracted points. In this section contours are detected using Moore neighbor algorithm. After extracting the contour from the binary image the palm and finger region area is selected.If second condition is satisfied that means if other finger is up excluding thumb finger then we proceeds the code to change the color. First we read the frame from the image and change the format of the frames from BGR to RGB.

```
if len(lmList) != 0:
  x1, y1 = lmList[8][1:]
x2, y2 = lmList[12][1:]
```

It gets the index 8s and 12s for x and y axis values and that is also skip the index values because it starts with value 1. Then check for pointing finger is up and thumb finger is down .If this condition is true then we performs two operations (1) converts width of the window relative to the screen width (2)converts height of the window relative to the screen height. And keep an eye on the pervious values of X and Y .autopy.mouse.move(wScr-cX, cY) It moves the mouse curser across the x3 and y3 values by inverting the direction.**if** finger[1] == 0 **and** finger[0] == 1: It checks to see if the pointer finger is down and thumb finger is up then it performs left click. After end of this process the obtained pixels are stored in an ordered array. These values are used for gesture detection. The following code is help to detect the color region:

```
check. img = cap.read()
imgRGB = cv2.cvtcolor(img. cv2.COLOR_BGR2RGB)
lmlist = handLandmarks(imgRGB)

if len(lmlist) !=0:
  x1, y1 = lmlist[0][1:]
  x2, y2 = lmlist[12][1:]
  x3 = numpy.interp(x1, (75, 640 - 75), (0, wScr))
  y3 = numpy.interp(y1, (75, 480 - 75), (0, hScr))

if results.multi_hand_landmarks:
for num, hand in enumerate(results.multi_hand_landmarks):
 mp_drawing.draw_landmarks(image, hand,
mp_hands.HAND_CONNECTIONS,mp_drawing.DrawingSpec(color=(121, 22, 76), thickness=2, circle_radius=4),
mp_drawing.DrawingSpec(color=(250, 44, 250),
thickness=2, circle_radius=2),
```

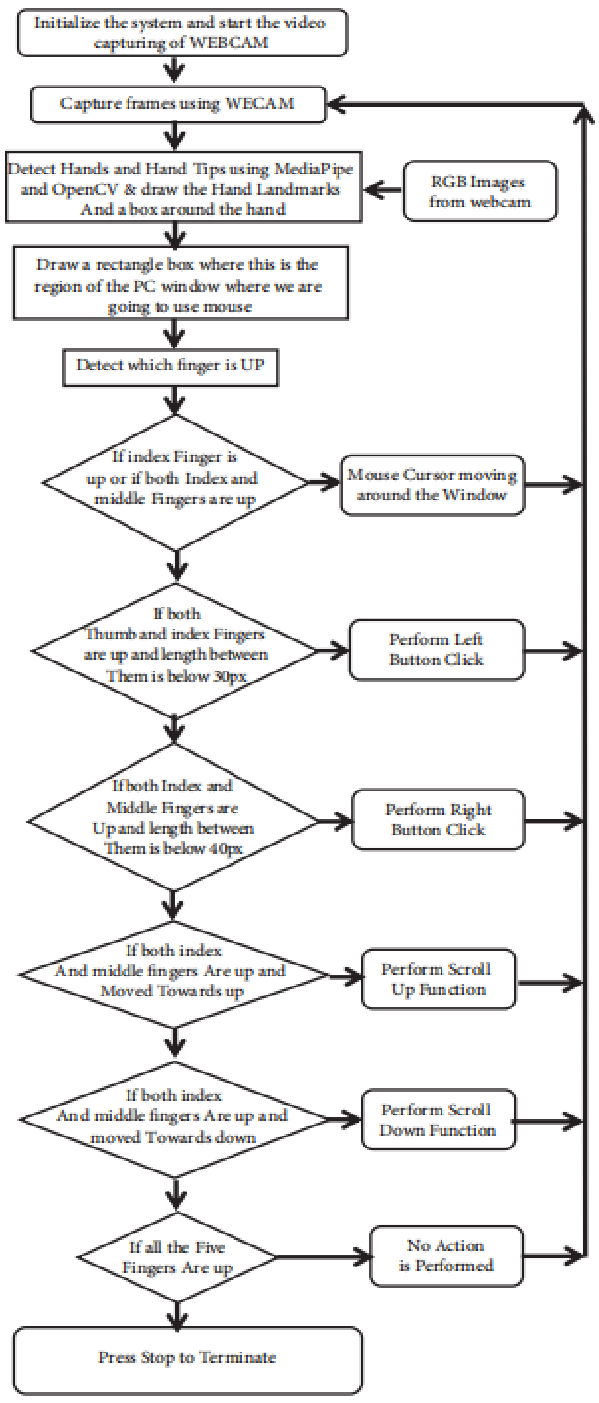
Initialize the system and start the video capturing of WEBCAM
Capture frames using WECAM
Detect Hands and Hand Tips using MediaPipe and OpenCV & draw the Hand Landmarks And a box around the hand
RGB Images from webcam
Draw a rectangle box where this is the region of the PC window where we are going to use mouse
Detect which finger is UP
If index Finger is up or if both Index and middle Fingers are up
Mouse Cursor moving around the Window
If both Thumb and index Fingers are up and length between Them is below 30px
Perform Left Button Click
If both Index and Middle Fingers are Up and length between Them is below 40px
Perform Right Button Click
If both index And middle fingers Are up and Moved Towards up
Perform Scroll Up Function
If both index And middle fingers Are up and moved Towards down
Perform Scroll Down Function
If all the Five Fingers Are up
No Action is Performed
Press Stop to Terminate

Fig. 4 : Flowchart of the proposed AI virtual mouse system

IV. EXPERIMENTAL RESULTS

In this paper, we are proposed a finger gesture based mouse system. The hand region of image frame and the center of the palm are first extracted from 3D structured image by the mediapipe framework. Then hand contours are extracted and show a graph line in 21 region point in hand. The autopy algorithm keeps tracks the finger movement and count the up finger by FingertipID python variable. Finally, to control the mouse cursor based on the finger location and their gesture. In our research we are considering the four function: cursor movement, left click, right click and scroll up-down. The fully explained working flow diagram of our proposed system is shown Figure 4.To explore working condition of gestures with real-time tracking, we investigate this system in some complicated cases, e.g., maximum distant from the camera, low light condition, dark or busy background during the tracking. The proposed can detect one person hand at a time .Unlike existing systems, this study uses only single CPU and does not require any external GPU system or special devices.

For performing cursor moving operationsif the index finger with Tipid=1 is up and middle perform no action, as shows in Figure 5.

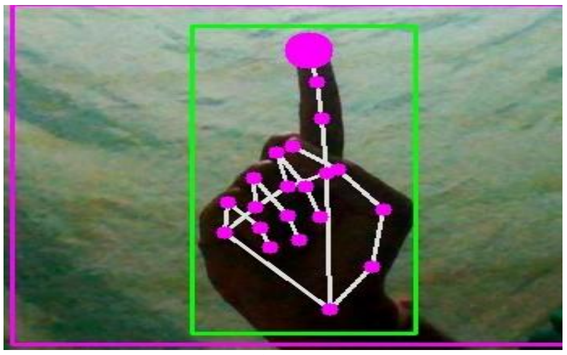
Fig. 5 : Gesture for cursor movement

For performing left click index finger with tipid=1 and middle finger with tipId=2 both are up and difference between two figure is 40pixles, as shows in Figure 6.

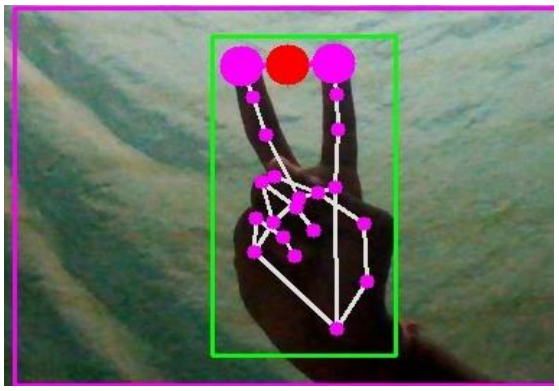
Fig. 6 : Gesture for left click

For performing right click if both the index finger tipid=1 and middle finger index id=2 are up and the distance between the two fingers is 40pixels, as shows in figure7.

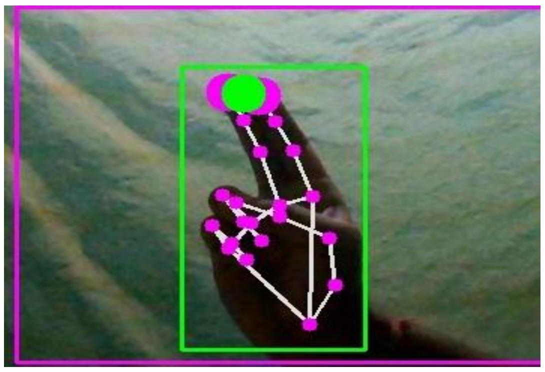
Fig. 7: Gesture for right click

For performing scroll up and downif the index finger with tipId=1 and middle finger with tipId=2 and distant between two finger is greater than 40Pixels, as shows in Figure 9.

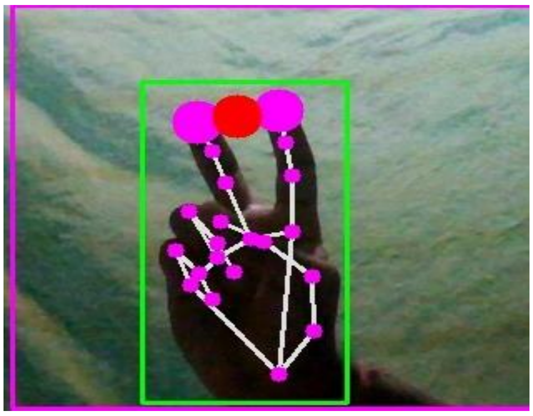
Fig. 8 : Gesture for scroll operation

We suggest the scroll operation with two finger for smooth operation but in this system we can scroll by three finger also. With three finger it is very hard to differentiate between two finger and three finger if the gestures are fast.

Finger Tip Gesture	Mouse Function Performed	Accuracy (%)
Tip Id 1 or both tip Id's 1 and 2 are up	Curser movement	100
Tip Id 0 and 1 are up and the distance between the fingers is <15	Left click	95
Tip Id 0 and 1 are up and the distance between the fingers is <20	Right click	98
Tip Id 0 and 1 are up and the distance between the fingers is >20 and both fingers are moved down	Scroll up	100
Tip Id 0 and 1 are up and the distance between the figures is <20 and both fingers are moved up	Scroll down	100
All five tip Id's 0,1,2,3,4 are up	No action	100
Result		100

Table1: Experimental results

Table 1 shows the experimental test results of our AI virtual mouse system. The results shows that there is no significant difference between physical mouse and virtual mouse operation. In our proposed system we can perform mouse operations with 100 percent accuracy .The results doesn't show any difference between fainted light and normal light. We shall first discuss the fingertip tracking preference in different lighting condition, noisy background and also in distant tracking conditions. In this analysis we record rapid gestures performance with their percentage of accuracy. In figure 9 the line graph shows the accuracy level of different performed actions in our virtual mouse system. In Figure10 our experimental results compared to another approaches using gestures for virtual mouse systems. But in our proposed system we can perform mouse operation with high accuracy.

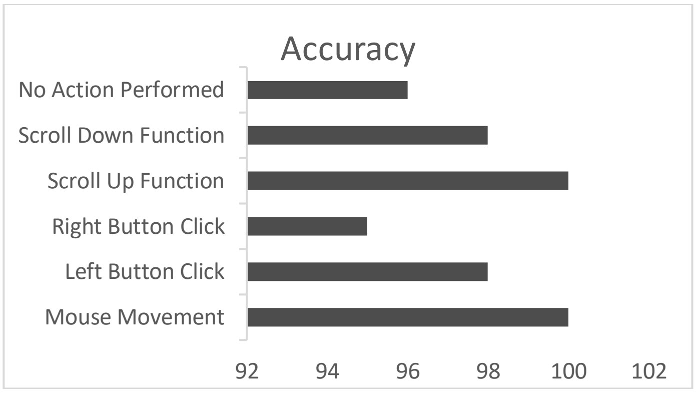

Fig. 9 :Graph of Accuracy for different performed actions

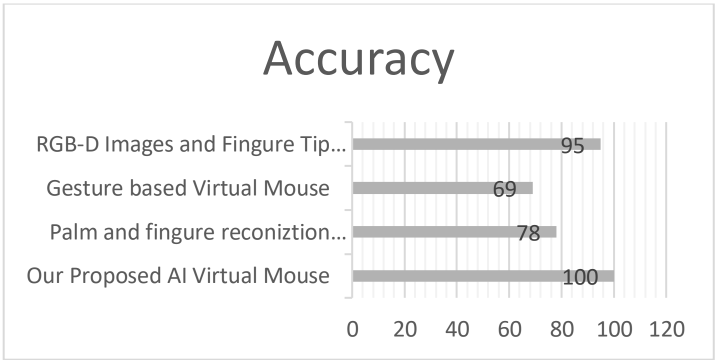

Fig. 10 : Accuracy graph fordifferentavailable mousesystem

V. APPLICATIONS

The AI virtual mouse system is beneficial for several applications like, it is utilized in situations where we cannot use the physical mouse and it will be accustomed reduce the space and value for using the physical mouse, and it improves the human-computer interaction. Major applications:

- The proposed model encompasses a accuracy of 99% which is way better than the that of other proposed models for virtual mouse.
- In COVID-19 situation, it's not so safe to use the physical devices by touching them because it should possible to make situation of spread the virus, so this proposed AI virtual mouse system may be wont to control the PC mouse functions without using the physical mouse.
- The system may be modified and wont to control automation and robots systems without usage of devices.
- AI virtual mouse will be accustomed play augmented reality and computer game based games without the wired or wireless mouse devices
- within the field of robotics, the proposed system like HCI are often used for controlling robots
- In the field of designing and architecture, this system will be used for designing virtually.

VI. CONCLUSION

This paper presented a brandnew AI virtual mouse method using OpenCv, autopy and mediapipe by the help of fingertip movement interacted with the computer in front of camera without using any physical device. The approach demonstrated high accuracy and highly accurate gesture eliminates in practical applications. The proposed method overcomes the limitations of already existing virtual mouse systems. It has many advantages, e.g., working well in low light as well as changing light condition with complex background, and fingertip tracking of different shape and size of finger with same accuracy. The experimental results shows that the approaching system can perform very well in real-time applications. We also intend to add new gesture on it to handle the system more easily and interact with other smart systems. It is possible to enrich the tracking system by using machine learning algorithm like Open pose. It is also possible to including body, hand and facial key points for different gestures.

REFERENCE

[1.] J. Katona, "A review of human–computer interaction and video game research fields in cognitive Info Communications," Applied Sciences, vol. 11, no. 6, p. 2646,2021.
[2.] D. L. Quam, "Gesture recognition with an information Glove," IEEE Conference on Aerospace and Electronics, vol. 2, pp. 755–760, 1990.
[3.] D.-H. Liou, D. Lee, and C.-C. Hsieh, "A real time hand gesture recognition system using motion history image," in Proceedings of the 2010 2nd International Conference on Signal Processing Systems, July 2010.
[4.] S. U. Dudhane, "Cursor system using hand gesture recognition," IJARCCE, vol. 2, no. 5, 2013.