Faculty of Engineering and Technology

University of Sindh Jamshoro

Group Members: Sameer 2K20/ITE/112

Muhammad Bux Buriro 2K20/ITE/80

Atif Buriro 2K20/ITE/91

Muhammad Awais 2K20/ITE/78

Hammad Arshad 2K20/ITE/46

Department: Information Technology(Evening)

Project: AI Virtual Mouse

Submitted to: Dr. Sandar Ali Khowaja

1. INTRODUCTION

In this high-tech world, it’s almost impossible to imagine life without computers. The invention of computers is one of the greatest humankind inventions. Computers have become an essential part of almost everyday use for individuals of every age. In daily life, we interact many times with computers to make our work easier. Thus, Human-Computer Interaction (HCI) has become a hot topic for research. In our daily life, vision and gestures are important approaches for communication among human beings, and the same role is played by the mouse in Graphical User Interface (GUI) based computers. So, a combined methodology can be used to make a better interactive system for Human-Computer Interaction. Computer vision techniques can be an alternative way for the touch screen and create a virtual human-computer interaction device using a webcam. In this project, a finger tracking-based virtual mouse application will be designed and implemented using a regular webcam. To implement this, we will be using the object tracking concept of Artificial Intelligence and the OpenCV module of Python.

1.1 Objective

The main objective of this project is as follows:

- To develop a virtual mouse using python programing language which works using hand gestures.
- To create a system which won’t require any hardware to operate mouse which will reduce the hardware cost.
- To find an alternative way to use at public places to reduce spread of virus through touch and will help people to return to normal routine after pandemic.

1.2 Limitations Of Existing System

- � The existing systems are our traditional hardware mouse devices either wired or wireless to control the cursor. This means the actual hardware device is required. Also, the touchpad in laptops and touch screen devices requires a user to touch the surface.
- Other existing virtual mouse control system consists of a simple mouse operation which uses colored tips like red, green, and blue color. These colored fingers act as an object that the web-cam senses to perform actions and then image processing techniques are applied to them.

- Some existing systems use number of fingers to perform the specific operations (for eg. 1 finger for left click, 2 fingers for right-click, 3 for double click). Such systems are more complex and difficult for the user to use.

1.3 Scope

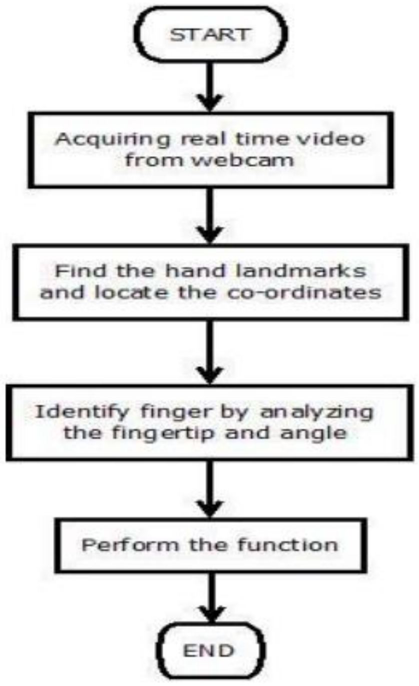
- The scope of this project is to develop a virtual mouse that will be operated without touching any device or screen. In today’s world where we are adjusting our living while being in a pandemic, a touch less mouse controller will be useful to eliminate the risk of spreading infection through touch on public service devices. Like a self-ticketing system at the railway station, many people touch the same screen which can increase the chances of virus infection.The virtual mouse can be used without touching the screen. This project can improve the scope of HumanComputer Interaction technology to be explored more.

2. PROPOSED SYSTEM

- There is no specific algorithm which we have used for virtual mouse but there are some pythons inbuild modules which will help in processing of this system. So the modules which we will be using are opencv, mediapipe, numpy, autopy, time, maths.

2.1 Design Details

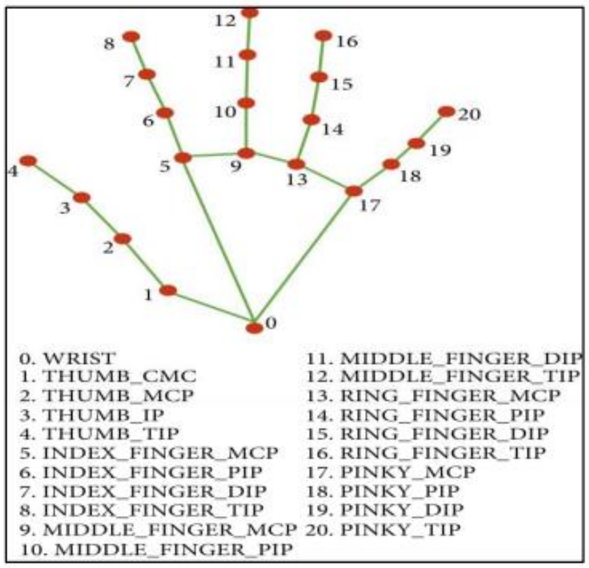
- The system will be taking the real-time video input from the webcam and converting it into array form to find the coordinates of the frame. These coordinates will help the program to locate the fingers in an accurate position and detect which fingers are raised. By identifying which finger is up system will perform certain functions assigned like moving or clicking.


START
Acquiring real time video from webcam
Find the hand landmarks and locate the co-ordinates
Identify finger by analyzing the fingertip and angle
Perform the function
END

2.2 Methodology

- Capturing video: The use of the OpenCV module is to capture the realtime video using a webcam which acts as an input for
further processing.

- Find hand landmarks Using Python libraries like CV2 and MediaPipe we coded the program to locate the hand landmarks. After recognizing hand it will locate 21 points as shown in the figure.

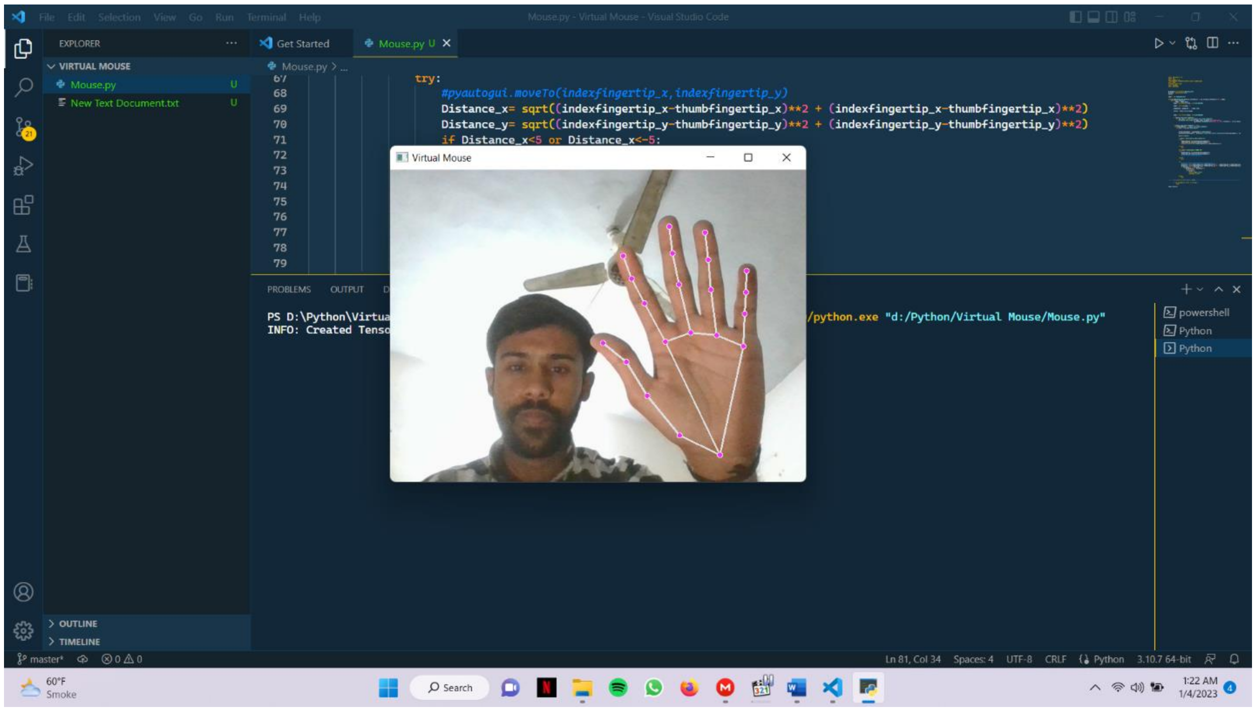


- Get the tip of the index and middle finger We also included different functions in a library named “HandTrackingModule” for simplicity. This module includes the functions for detecting hands, locating fingers, counting which fingers are up, finding the distance between fingers, etc.
- Check which fingers are up Program checks which fingers are up.
- **Index fingers:** If only the index finger is up that means the mouse is in moving mode. Users can move their finger to move the cursor on the screen.
- **Index finger and Middle finger:** If both of these fingers are up then the mouse is in clicking mode. When the distance between these two fingers is short, the clicking function will be performed.
- **Convert the coordinates to get the correct positioning**
  Accurate location tracking of the cursor on the screen is important for the exact working of the mouse.
- **Implement the functions**
  By locating the coordinates and tracking the fingers we perform the mouse functions virtually i.e. without any physical contact with the device.
- **Frame rate**
  Frame rate helps us to check if the movements of the cursor are smooth or not.
- **Smoothen the values so the mouse is not jittery**
  By observing the change in frame rate and movement of cursor we applied some smoothing technique to mouse so that it is easy to use for user.
- **Display**
  We also displayed the tracking by webcam to show implementation properly.

Code:

```
import mediapipe as mp
import cv2
import numpy as np
from mediapipe.framework.formats import landmark_pb2
import time
from math import sqrt
import win32api
import pyautogui

mp_drawing = mp.solutions.drawing_utils
mp_hands = mp.solutions.hands
click=0

video = cv2.VideoCapture(0)

with mp_hands.Hands(min_detection_confidence=0.8, min_tracking_confidence=0.8) as hands:
    while video.isOpened():
        _, frame = video.read()
        image = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

        image = cv2.flip(image, 1)

        imageHeight, imageWidth, _ = image.shape

        results = hands.process(image)


        image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)

        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=(250, 44, 250), thickness=2, circle_radius=2),
                                          )

        if results.multi_hand_landmarks != None:
          for handLandmarks in results.multi_hand_landmarks:
            for point in mp_hands.HandLandmark:


                normalizedLandmark = handLandmarks.landmark[point]
            pixelCoordinatesLandmark =
mp_drawing._normalized_to_pixel_coordinates(normalizedLandmark.x, normalizedLandmark.y,
imageWidth, imageHeight)

            point=str(point)

            if point=='HandLandmark.INDEX_FINGER_TIP':
             try:
                indexfingertip_x=pixelCoordinatesLandmark[0]
                indexfingertip_y=pixelCoordinatesLandmark[1]
                win32api.SetCursorPos((indexfingertip_x*4,indexfingertip_y*5))

             except:
                pass

            elif point=='HandLandmark.THUMB_TIP':
             try:
                thumbfingertip_x=pixelCoordinatesLandmark[0]
                thumbfingertip_y=pixelCoordinatesLandmark[1]
                #print("thumb",thumbfingertip_x)

             except:
                pass

            try:
                #pyautogui.moveTo(indexfingertip_x,indexfingertip_y)
                Distance_x= sqrt((indexfingertip_x-thumbfingertip_x)**2 + (indexfingertip_x-
thumbfingertip_x)**2)
                Distance_y= sqrt((indexfingertip_y-thumbfingertip_y)**2 + (indexfingertip_y-
thumbfingertip_y)**2)
                if Distance_x<5 or Distance_x<-5:
                    if Distance_y<5 or Distance_y<-5:
                        click=click+1
                        if click%5==0:
                            print("single click")
                            pyautogui.click()

            except:
                pass

        cv2.imshow('Virtual Mouse', image)
        if cv2.waitKey(10) & 0xFF == ord('q'):
            break
video.release()
```

Output:



```
            try:
                #pyautogui.moveTo(indexfingertip_x,indexfingertip_y)
                Distance_x= sqrt((indexfingertip_x-thumbfingertip_x)**2 + (indexfingertip_x-thumbfingertip_x)**2)
                Distance_y= sqrt((indexfingertip_y-thumbfingertip_y)**2 + (indexfingertip_y-thumbfingertip_y)**2)
                if Distance_x<5 or Distance_x<-5:
```

```
PS D:\Python\Virtua                                   /python.exe "d:/Python/Virtual Mouse/Mouse.py"
INFO: Created Tenso
```

Github Link: https://github.com/sameerpanhwarit/Virtual-Mouse