TATYASAHEB KORE INSTITUTE OF ENGINEERING AND TECHNOLOGY, WARANANAGAR

(AN AUTONOMOUS INSTITUTE)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

SHREE WARANA VIBHAG SHIKSHAN MANDAL
TATYASAHEB KORE INSTITUTE OF ENGINEERING & TECHNOLOGY
WARANANAGAR
PERFECTION EFFICIENCY PRECISION
SWVSM'S
TKIET

A DOMAIN SPECIFIC MINI PROJECT REPORT ON

"Gesture recognition for Human Computer Interaction"

Submitted By

GROUP NAME: TEA18

Name	Roll No
Mr. Gaikwad Nilesh Chandrakant	37
Mr. Ghule Pranav Arjun	44
Mr. Gurav Jayash Uday	47
Mr. Parit Swaroop Kiran	67
Mr. Ganesh Ananda Khot	69

Students of Third Year Computer Science & Engineering.

Under the Guidance Of

Prof A. S. Phalle

Associate Professor,
Department of Computer science and Engineering
Tatyasaheb Kore Institute of Engineering and Technology,
Warananagar

ACADEMIC YEAR : 2021-2022

TATYASAHEB KORE INSTITUTE OF ENGINEERING AND TECHNOLOGY,

WARANANAGAR

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that, Domain Specific Mini Project report entitled, "**Gesture recognition for Human Computer Interaction**" submitted by

Student Name	Exam Seat No	Roll No.
Mr.Nilesh Chandrakant Gaikwad	4881	37
Mr.Pranav Arjun Ghule	4891	44
Mr.Jayash Uday Gurav	4896	47
Mr.Swaroop Kiran Parit	4972	67
Mr.Ganesh Ananda Khot	4929	69

of third year Computer Science and Engineering TKIET, Warananagar, in partial fulfillment for the award of bachelor degree in Computer Science and Engineering from Shivaji University, Kolhapur. It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report. This Domain Specific Mini Project work is a record of students own work, carried out by them under our supervision and guidance during academic year 2021-2022.

Prof.A.S.Phalle
(Project Guide)

Dr.G.V.Patil
(Head of Department)

Dr. S. V. Anekar
(Principal)

.....
(Internal Examiner)

.....
(External Examiner)

Place:Warananagar

Date: _____

ACKNOWLEDGMENT

This space is dedicated to acknowledge all those who have helped in bringing this Domain Specific Mini Project to fruition.

We are greatly indebted to our guide **Prof A.S.Phalle**, for his unstinted support and valuable suggestions. We are grateful to him not only for the guidance, but also for his unending patience and keeping our spirits high throughout. We express our sincere thanks to our beloved Head of the Department, **Dr**.**G.V.Patil** and Principal, **Dr. S. V. Anekar** for being a source of inspiration and providing us the opportunity to work on this project.

We extend heartfelt thanks to all the teaching and non-teaching staff of the department of Computer Science and Engineering for their assistance and cooperation.

Finally, we would like to thank our parents and friends for their moral support and encouragement throughout our academics.

Student Name	Exam Seat No	Sign
Mr.Nilesh Chandrakant Gaikwad	4881	
Mr.Pranav Arjun Ghule	4891	
Mr.Jayash Uday Gurav	4896	
Mr.Swaroop Kiran Parit	4972	
Mr.Ganesh Ananda Khot	4929	

Date: **Place: Warananagar**

Content

Chapter 1
INTRODUCTION

1.1 Domain Introduction

Computer vision is an interdisciplinary scientific field that deals with how computers can gain high-level understanding from digital images or videos. From the perspective of engineering, it seeks to understand and automate tasks that the human visual system can do.

Computer vision tasks include methods for acquiring, processing, analyzing and understanding digital images, and extraction of high-dimensional data from the real world in order to produce numerical or symbolic information, e.g. in the forms of decisions Understanding in this context means the transformation of visual images (the input of the retina) into descriptions of the world that make sense to thought processes and can elicit appropriate action. This image understanding can be seen as the disentangling of symbolic information from image data using models constructed with the aid of geometry, physics, statistics, and learning theory.

The scientific discipline of computer vision is concerned with the theory behind artificial systems that extract information from images. The image data can take many forms, such as video sequences, views from multiple cameras, multi-dimensional data from a 3D scanner, or medical scanning device. The technological discipline of computer vision seeks to apply its theories and models to the construction of computer vision systems.

Sub-domains of computer vision include scene reconstruction, object detection, event detection, video tracking, object recognition, 3D pose estimation, learning, indexing, motion estimation, visual servoing, 3D scene modeling, and image restoration.

1.2 Existing System

"Hand Gesture Recognition Using Camera" is based on concept of Image processing. In recent year there is lot of research on gesture recognition using HD camera but cameras are more costly. This project is focus on reduce cost and improve robustness of the proposed system using simple web camera.

1.3 Drawbacks of Existing System

Unfortunately, like other image processing problems, hand tracking and segmentation in a cluttered background is a critical problem in hand gesture recognition. In most occasions, the background is not simple. Also, the background may contain other body parts captured by the camera.

1.4 Motivation

hand gestures along with their assigned commands (functions) that are used in the complex environment. Though the gestures have to be mapped for commands in specific applications, the same gesture vocabulary could be reused for mapping different set of commands according to different range of applications like controlling games, browsing images etc. This makes the gesture recognition system more generalized and adaptive towards human computer interaction.

Chapter 2

PROBLEM DEFINITION AND SCOPE

` 2.1 Problem statement

"Hand Gesture Recognition Using Camera" is based on concept of Image processing. In recent year there is lot of research on gesture recognition using HD camera but cameras are more costly. This project is focus on reduce cost and improve robustness of the proposed system using simple web camera.

2.2 Objectives

In a day-to-day life, hand gesture recognition is one of the system that can detect the gesture of hand in a real time video. The gesture of hand is classify within a certain area of interest. Designing a system for hand gesture recognition is one of the goal of achieving the objectives of this project. The task of recognizing hand gestures is one of the main and important issues in computer vision. With the latest advances in information and media technology, human computer interaction (HCI) systems that involve hand processing tasks such as hand detection and hand gesture recognition.

2.3 Technologies and Associated Platforms

2.3.1 Hardware Requirements

- Laptop / PC
- Minimum i3 core processor
- Webcam Connection
- Minimum 32Byte system

2.3.2 Software Requirements

- Python Compiler
- Pycharm IDE
- pyautogui==0.9.53
- opencv-python==4.5.3.56
- mediapipe==0.8.6.2

Chapter 3

REQUIREMENT ANALYSIS

3.1 Functional Requirements

3.1.1 Camera GUI :

As we need to capture the frames of hand gestures , we need to actually implement it through a Camera GUI window. This can be done using OpenCV.

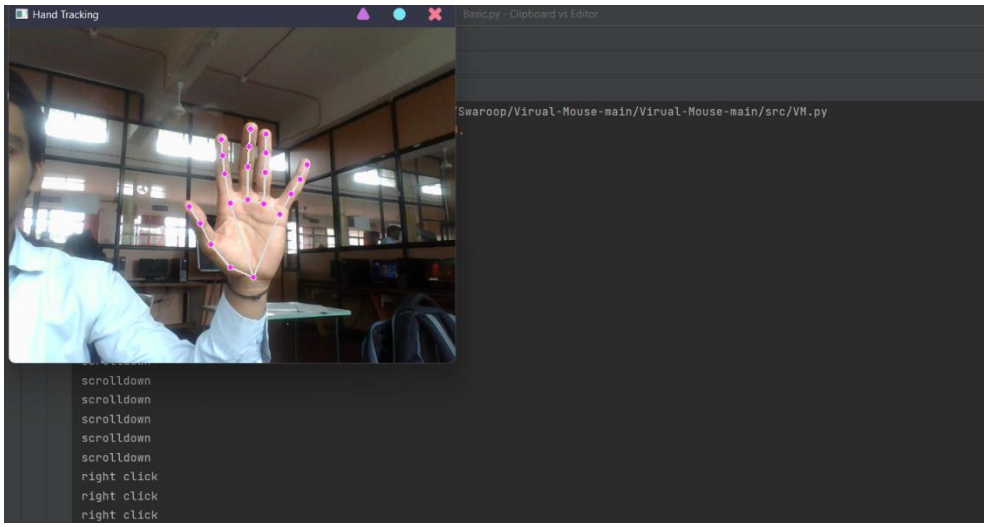


Fig.3.1.2 Camera GUI

3.1.2 Hand joint detection using OpenCV and MediaPipe

MediaPipe is a cross-platform framework for building multimodal applied machine learning pipelines. MediaPipe Python package is available on PyPI for Linux, macOS, and Windows.

a simple code for hand joint detection using OpenCV.

At first, import the necessary packages –

```
import cv2
import mediapipe as mp
import time
```

Initialize the classes-

```
cap = cv2.VideoCapture(0)
mpHands = mp.solutions.hands
hands = mpHands.Hands()
mpDraw =
mp.solutions.drawing_utils
```

Final code-

```
pTime = 0
cTime = 0

while True:
    success, image = cap.read()
    imgRGB = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    results = hands.process(imgRGB)
    #rint(results.multi_hand_landmarks)
    if results.multi_hand_landmarks:
        for handLms in results.multi_hand_landmarks:
            for id, lm in enumerate (handLms.landmark):
                #print(id,lm)
                h, w, c = image.shape
                cx, cy = int(lm.x*w), int(lm.y*h)
                print(id, cx, cy)
                if id == 4:
                    cv2.circle(image,(cx,cy),15,
(255,0,255),cv2.FILLED)
            mpDraw.draw_landmarks(image, handLms,
mpHands.HAND_CONNECTIONS)
    cTime = time.time()
    fps = 1/(cTime-pTime)
    pTime = cTime
```

```
    cv2.putText(image, str(int(fps)),(10,60),
cv2.FONT_HERSHEY_PLAIN,3, (255,0,255),4)
    cv2.imshow("Results", image)
    if cv2.waitKey(1) & 0XFF == ord('q'):
        break
```

Output -

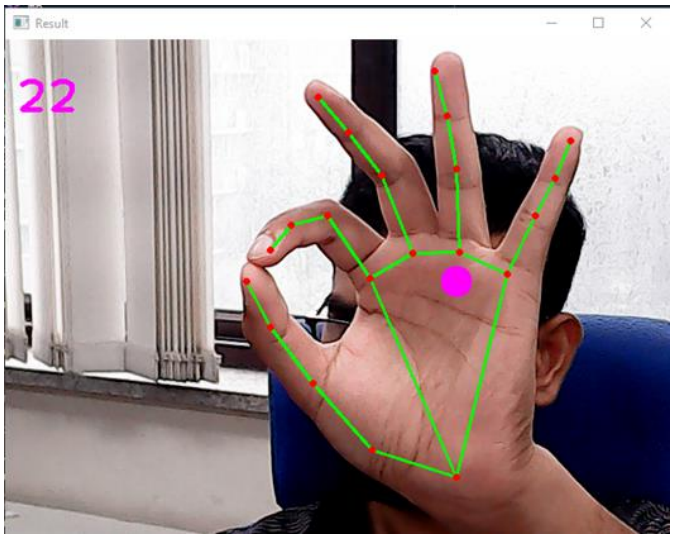


Fig.3.1.1 Hand Mappings

3.1.3 PyautoGUI :

- pyautogui==0.9.53

  This version library is must be used to control the cursor movement. This library can be used to connect hand mappings to gestures.

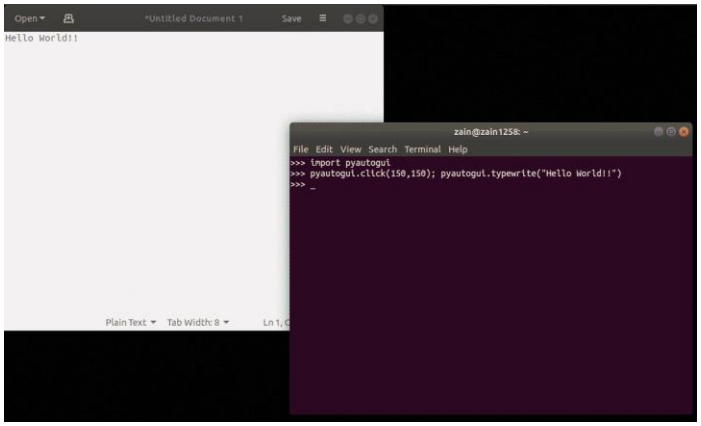


Fig 3.1.3 PyautoGUI

3.1.4 Some Important Libraries :

1.OpenCV :

OpenCV is a great tool for image processing and performing computer vision tasks. It is an open-source library that can be used to perform tasks like face detection, objection tracking, landmark detection, and much more. It supports multiple languages including python, java C++.

2.Mediapipe:

Mediapipe is a cross-platform library developed by Google that provides amazing ready-to-use ML solutions for computer vision tasks. OpenCV library in python is a computer vision library that is widely used for image analysis, image processing, detection, recognition, etc.

3.Autopy:

AutoPy is a simple, cross-platform GUI automation library for Python. It includes functions for controlling the keyboard and mouse, finding colors and bitmaps on-screen, and displaying alerts.

4.PyautoGui:

This python library is used for controlling the movement of Mouse and Keyboard . We Can control movements using set of code.

5.C-Types :

Used for c type variable or method initialization .

6.Math :

For basic mathematical formulas and calculations

Chapter 4

DESIGN METHODOLOGY

4.1 System Architecture

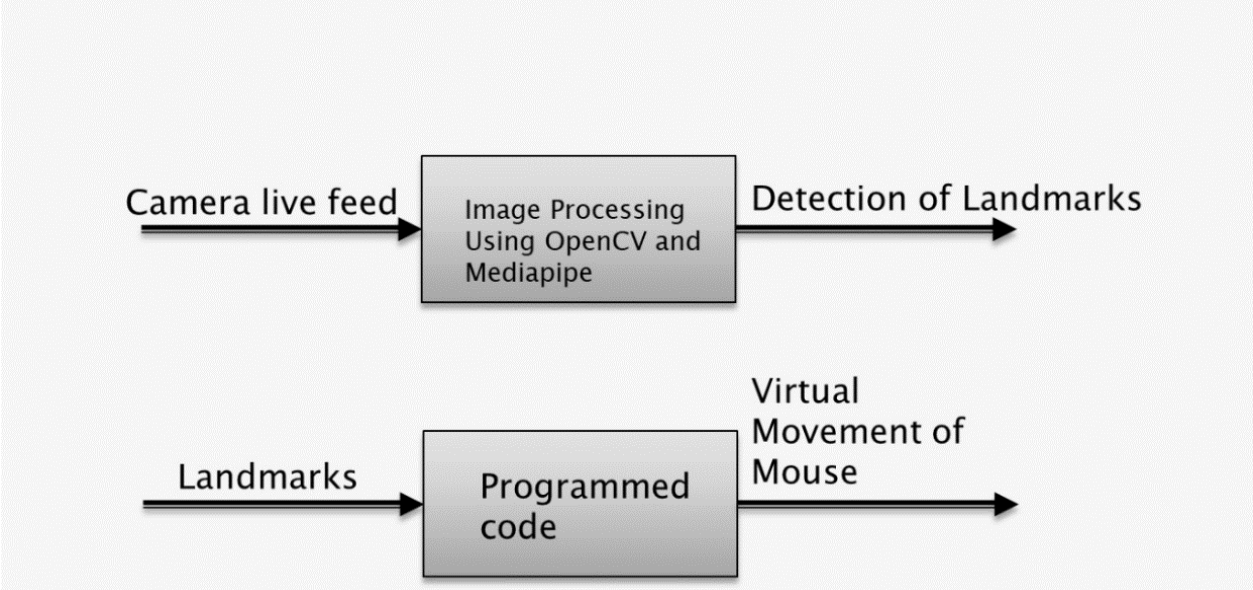


Fig 4.1: System Architecture

Camera Live Feed Recognize the hand position as input And using Libraries (OpenCV and MediaPipe) Detects the hand land mark marks. Mappings are done and are connected to mouse movement by using PyAutoGUI library.

Rectangular Region for Moving through the Window. The AI virtual mouse system makes use of the transformational algorithm, and it converts the coordinates of fingertip from the webcam screen to the computer window full screen for controlling the mouse. When the hands are detected and when we find which finger is up for performing the specific mouse function, a rectangular box is drawn with respect to the computer window in the webcam region where we move throughout the window using the mouse cursor.After image processing through code , program perform the Virtual Mouse moment. The Mouse movement is based on various gestures assigned on hand.

4.2 Flow Chart:

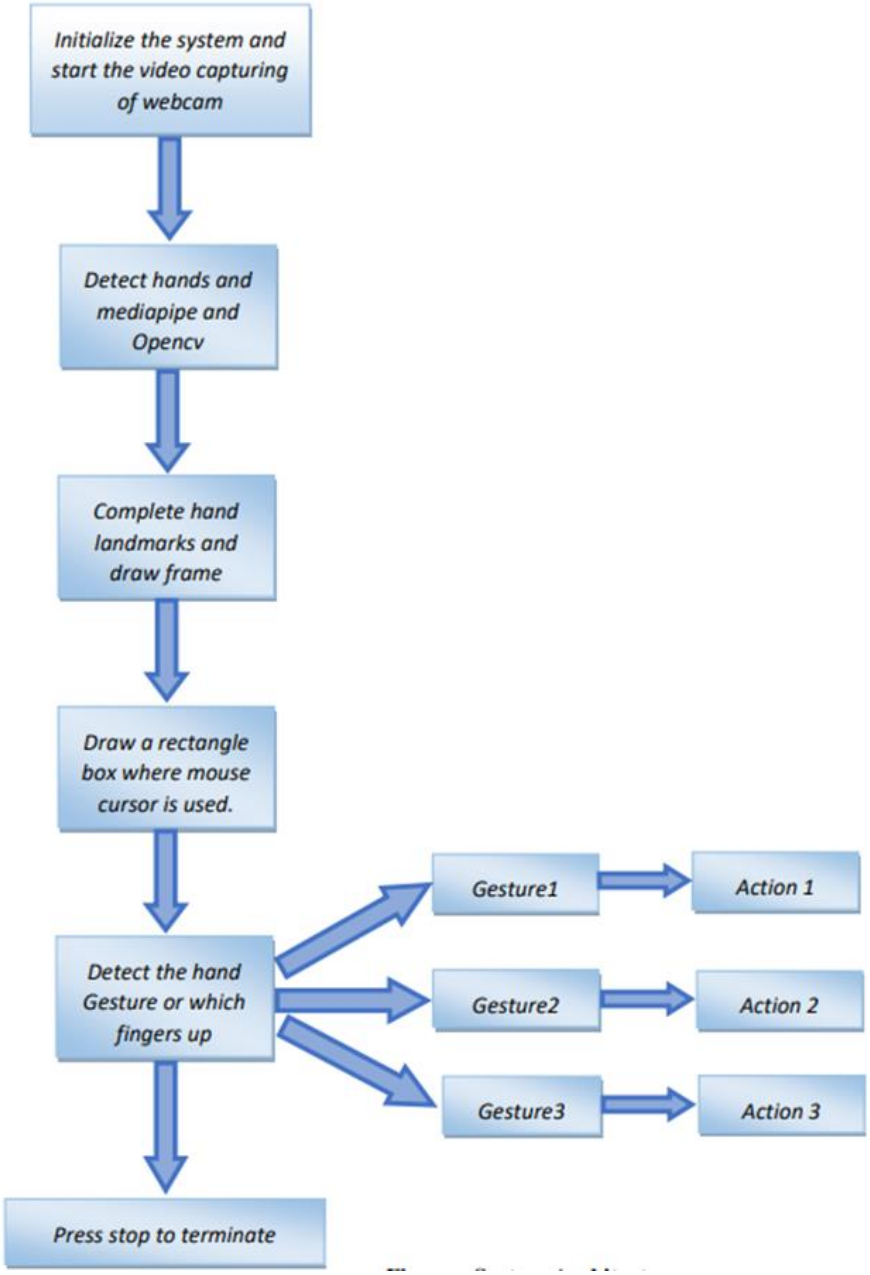


Fig 4.2 : System Flow Diagram

As processing is done in many of the stages there can be association of many gestures to perform various actions. Camera Live Feed Recognize the hand position as input
And using Libraries (OpenCV and MediaPipe) Detects the hand land mark marks. Mappings are done and are connected to mouse movement by using PyAutoGUI library.

After image processing through code , program perform the Virtual Mouse moment. The Mouse movement is based on various gestures assigned on hand.

Chapter 5

System Design and Implementation

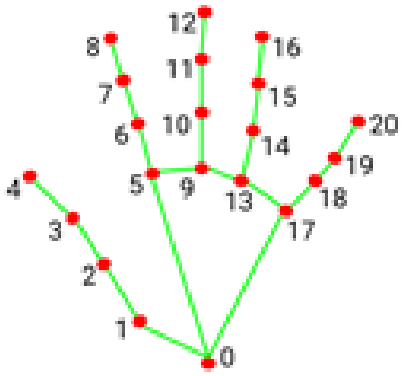
5.1 Methology

The various functions and conditions used in the system are explained in the flowchart of the real-time AI virtual mouse system in figure.

Camera Used in the AI Virtual Mouse System. The proposed AI virtual mouse system is based on the frames that have been captured by the webcam in a laptop or PC. By using the Python computer vision library OpenCV, the video capture object is created and the web camera will start capturing video, as shown in Figure. The web camera captures and passes the frames to the AI virtual system.

Capturing the Video and Processing. The AI virtual mouse system uses the webcam where each frame is captured till the termination of the program. The video frames are processed from BGR to RGB colour space to find the hands in the video frame by frame as shown in the following code:

```
def findHands(self, img , draw = True):
imgRGB = cv2.cvtColor(img , cv2.COLOR_BGR2RGB)
 self.results = self.hands.process(imgRGB)
```



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

Rectangular Region for Moving through the Window. The AI virtual mouse system makes use of the transformational algorithm, and it converts the coordinates of fingertip from the webcam screen to the computer window full screen for controlling the mouse. When the hands are detected and when we find which finger is up for performing the specific mouse function, a rectangular box is drawn with respect to the computer window in the webcam region where we move throughout the window using the mouse cursor.

5.2 Module 1: (Double Click)

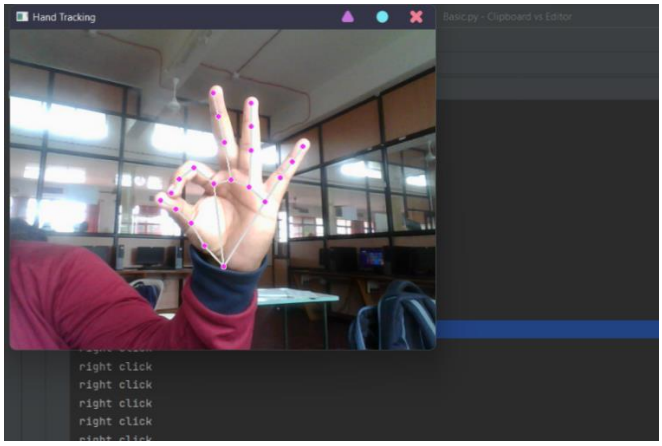


Fig 6.1.1 : Double click

```
try:
        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("double click")
              pyautogui.click(clicks = 2)
    except:
    pass
```

This module performs Double click when distance between index finger and thumb is lesser than integer 5.

5.2 Module 2: (Right Click)

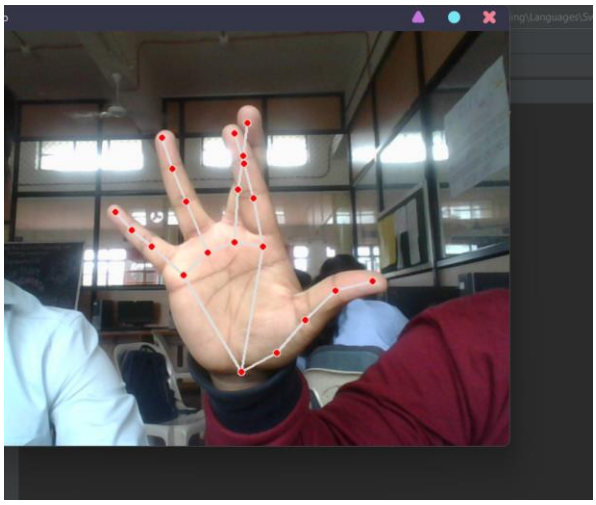


Fig 6.1.2 : Right click

```
try:
        Distance_x = sqrt(
          (indexfingertip_x - middlefingertip_x) * 2 + (indexfingertip_x -
middlefingertip_x) * 2)
        Distance_y = sqrt(
          (indexfingertip_y - middlefingertip_y) * 2 + (indexfingertip_y -
middlefingertip_y) * 2)
        if Distance_x < 5 or Distance_x < -5:
          if Distance_y < 5 or Distance_y < -5:
            click = click + 1
            if click % 2 == 0:
              print("right click")
              pyautogui.click(button='right')

    except:
      pass
```

This module performs right click when distance between index finger and middle finger is lesser than integer 5.

5.2 Module 3: (Scroll Up)

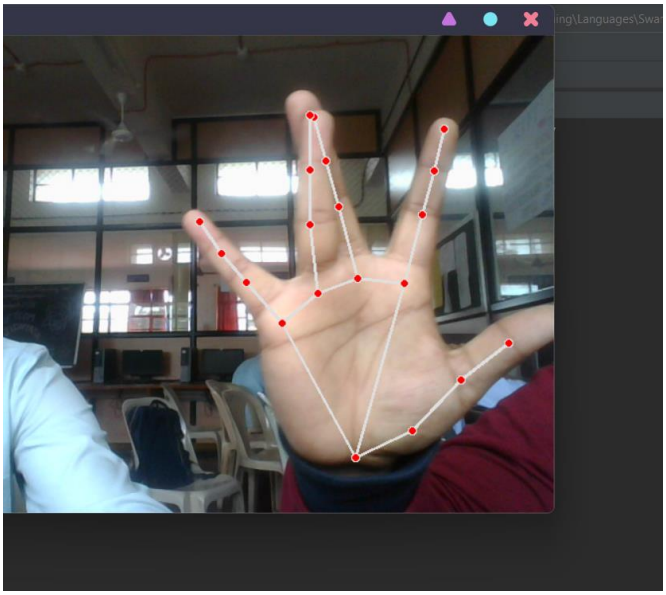


Fig 6.1.3 : Scroll Up

```
try:
        Distance_x = sqrt((ringfingertip_x - middlefingertip_x) * 2 +
(ringfingertip_x - middlefingertip_x) * 2)
        Distance_y = sqrt((ringfingertip_y - middlefingertip_y) * 2 +
(ringfingertip_y - middlefingertip_y) * 2)
        if Distance_x < 5 or Distance_x < -5:
          if Distance_y < 5 or Distance_y < -5:
            click = click + 1
            if click % 2 == 0:
              print("scrollup")
              pyautogui.vscroll(30)
    except:
      pass
```

This module performs Scroll up when distance between Ring finger and Middle finger is lesser than integer 5.

5.2 Module 4: (Scroll Down)

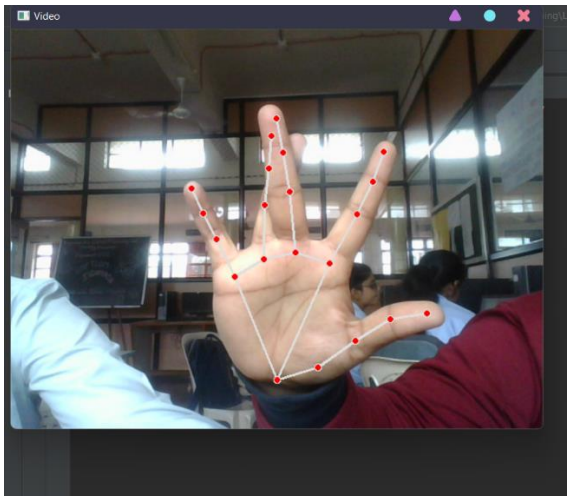


Fig 6.1.3 : Scroll Down

```
try:
        Distance_x = sqrt((middlefingertip_x - ringfingertip_x) * 2 +
(middlefingertip_x - ringfingertip_x) * 2)
        Distance_y = sqrt((middlefingertip_y - ringfingertip_y) * 2 +
(middlefingertip_y - ringfingertip_y) * 2)
        if Distance_x < 5 or Distance_x < -5:
            if Distance_y < 5 or Distance_y < -5:
                click = click + 1
                if click % 2 == 0:
                    print("scrolldown")
                    pyautogui.vscroll(-30)
    except:
        pass
```

This module performs Scroll up when distance between Ring finger and Middle finger is lesser than integer 5.

Chapter 6

CONCLUSION & FUTURE SCOPE

Due to accuracy and efficiency plays an important role in making the program as — useful as an actual physical mouse, a few techniques had to be implemented. After implanting such type of application there is big replacement of physical mouse i.e., there is no need of any physical mouse. Each & every movement of physical mouse is done with this motion tracking mouse (virtual mouse).

- There are several features and improvements needed in order for the program to be more user friendly, accurate, and flexible in various environments. The following describes the improvements and the features required:
- a) Smart Movement: Due to the current recognition process are limited within 25cm radius, an adaptive zoom in/out functions are required to improve the covered distance, where it can automatically adjust the focus rate based on the distance between the users and the webcam.
- b) Better Accuracy & Performance: The response time are heavily relying on the hardware of the machine, this includes the processing speed of the processor, the size of the available RAM, and the available features of webcam. Therefore, the program may have better performance when it's running on a decent machine with a webcam that performs better in different types of lightings.
- c) Mobile Application: In future this web application also able to use on Android devices, where touchscreen concept is replaced by hand gestures.

Chapter 7

REFERENCES

1. OpenCV Website – www.opencv.org
2. Murtaza workshop –murtazahassan.com
3. Advanced Computer Vision (Free code camp) - https://www.youtube.com/watch?