



RAMAIAH
Institute of Technology

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

RAMAIAH INSTITUTE OF TECHNOLOGY
(AUTONOMOUS INSTITUTE AFFILIATED TO VTU)
M. S. R. I. T. POST, BANGALORE – 560054
2022-2023

Mini Project Report

Controlling Mouse Motions Using Eye Tracking

Using Computer Vision

for the subject

Computer Vision (ISE555)

In

Fifth Semester

Submitted By:

Name:

JATIN B

PANNAGA N

SWAMY NAGA GANA TEJA C H

SANJEEV G

USN:

1MS20IS054

1MS20IS083

1MS20IS121

1MS21IS406

Submitted To:

Dr. Megha P Arakeri

Associate Professor,

Dept.of ISE,RIT

CONTENT

1. Abstract.....	3
2. Introduction.....	4
3. Methodology.....	5
4. Implementation.....	8
5. Code.....	11
6. Results.....	14
7. Project schedule.....	18
8. References.....	19

Abstract

This paper presents a novel algorithm for controlling the movement of a computer screen cursor using the iris movement. By accurately detecting the position of the iris in the eye and mapping that to a specific position on the computer screen. Which advances in eye-tracking technology have led to better human-computer interaction, and involve controlling a computer without any kind of physical contact. This research describes the transformation of a commercial eye-tracker for use as an alternative peripheral device in human-computer interactions, implementing a pointer that only needs the eye movements of a user facing a computer screen, thus replacing the need to control the software by hand movements. The algorithm enables physically disabled individuals to control the computer cursor movement to the left, right, up and down. The algorithm also enables the person to open and close folders or files or applications through a clicking mechanism.

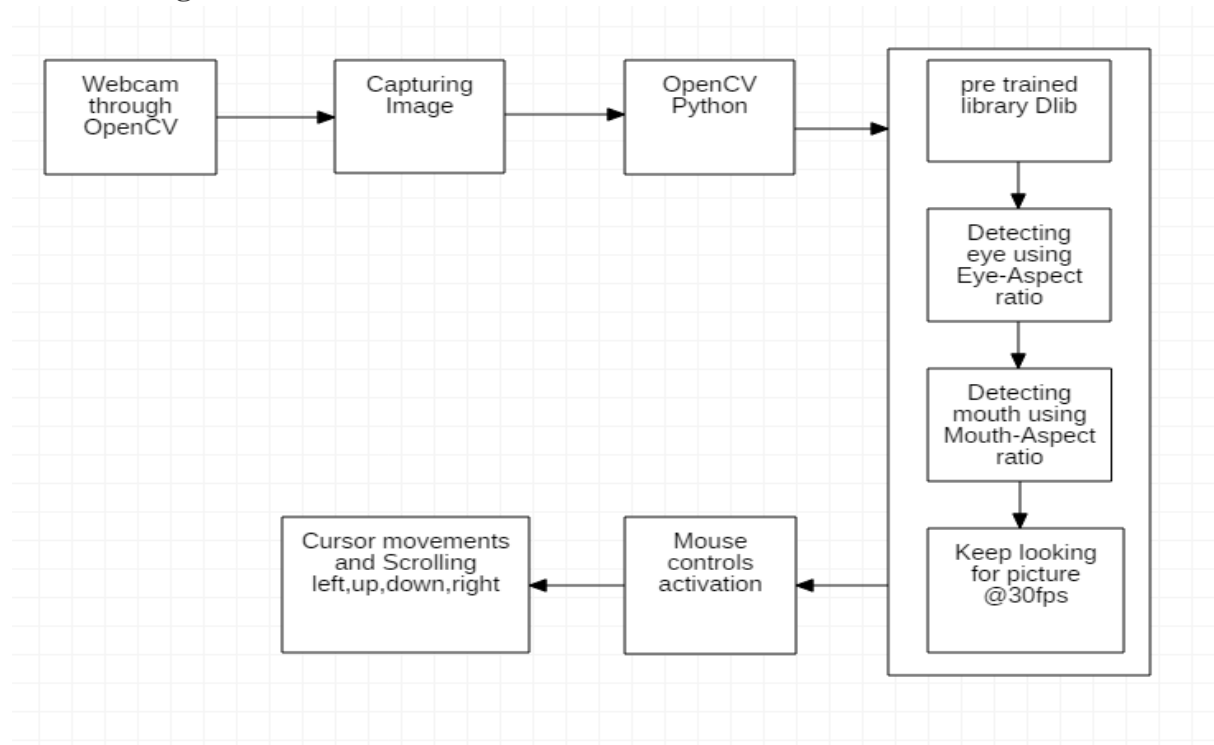
INTRODUCTION

Computers arrived in 1938 to comfortably improve our lives. There have been stringent improvements. They first worked hard to solve problems with mathematics and word processing and now have made a long way to make our lives a necessary part of their professional and personal needs, such as Internet surfing, Connect the globe through social media. In due time, operations on Computers can simply be done by people with hands and eyes and can operate mouse and keyboard and also can see, what's on the screen. The computer with the speech text alternative is now used by blind people which speak-out the layout on the screen. But people who don't have hands can't use it because of its mouse and mouse features That's why they are not able to compete with modernizing and changing world. By introducing technology and making them compatible to their computers, they don't have to get down from their dreams and expectations because of disability, that they can learn and live. Professionals work hard to help disabled people to use signals such as brain electroencephalography, EMG, facial muscle signals and electrooculogram tool to interact with computers. They are (EOG). In addition, methods such as limbus, tracking of eye and detection of blink, contact lens method, corneal related things and reflection of pupil are also used. Operations with keywords. In the competitive environment of today. All must be equipped with various skill sets. Well, just like get a job. Computers are a knowledge and use nowadays, necessity. The use of attachments and electrodes are needed in these methods, so the head is not physically connected. There are no such expensive equipment in the method described in this paper. Also, every equipment needs to make even physical connections with the user. It just needs a webcam computer, making it a simple, inexpensive and straightforward way. By capturing real-time user pictures, it is intended to evaluate these pictures and to identify the action type by matching the previously stored phrases to recognize the procedure to be performed and, when necessary, parameter when the cursor is moved. The application with proposed method would offer people with no hands a wide range of options.

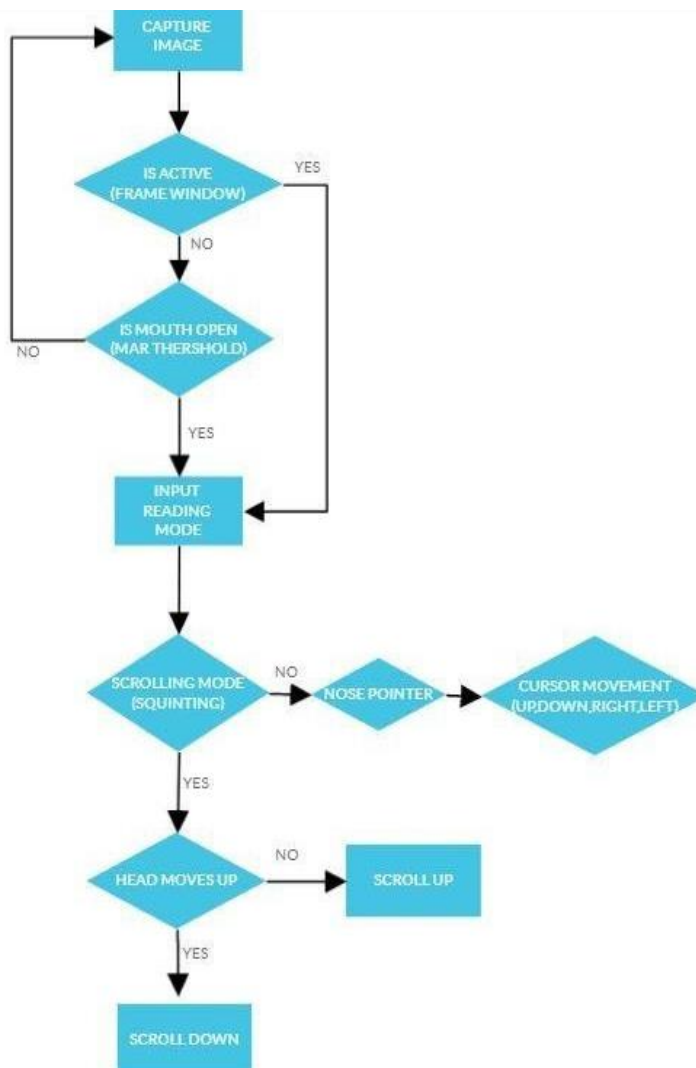
Methodology

In this section, we will discuss about the proposed approach to implement the mouse cursor detection using eye and face gestures. The process repeats to detect the face of a person using the computer vision and shape predicting library. The location of the mouth and eyes is registered until the face is identified, to monitor functions of the mouse, such as left clicking and moving cursors. For this system, no special equipment and sensors are needed. It is a hands-free machine for handicapped persons. You must first turn on mouse control mode by opening your mouth. To move the cursor to appropriate pointed direction, we only move the head in the direction the cursor needs to be moved. We need to turn the eyes in the camera to activate scroll mode and it shows scrollmode on when you start. We can move our head up and down to scroll the page

I. Block Diagram:



II. Flow Chart:



III. Algorithm:

Step-1: As soon as we start the application the desktop raises a frame window and the camera starts capturing the image and the frame window displays the video capturing that is being captured by the camera.

Step-2: Then Shape Predictor will try to landmark the co-ordinate points on the face and the continuous capturing of the video will depict the changes being made spontaneously.

Step-3: In order to read the input, the application will check for change in mouth aspect ratio, if it is more than the threshold then the frame will start reading the input.

Step-4: In input reading mode, we will have a rectangle around the nose and the nose point movements make the cursor move up, down, left and right.

Step-5: In the reading mode it will also check the change in Eye aspect ratio, if it is below the threshold for either of the eyes then that click is made (right-click, left-click).

Step-6: If the Eye aspect ratio of both the eyes is below the threshold (squint) then, it is considered as scrolling mode for scrolling up and down to navigate through documents.

Step-7: If we want to stop the input reading mode, the mouth aspect ratio has to meet the threshold again.

Implementation

The method for mouse movement by pupil and mouth movement was achieved using the following process: -

A. Face Detection: -

To represent a clean and accurate image, the user should face parallel to the webcam position as represented in Fig1. The image of the person is captured by webcam preinstalled in the system using OpenCV and processed using python. Fig2 shows the captured image using a webcam.

Dlib has a pre-trained dataset iBUG 300-W which has 68 coordinates that mapped to the face of a person. The facial detector detects key landmarks on the face of a person and tracks them. Fig. 3 shows 68 facial coordinates.

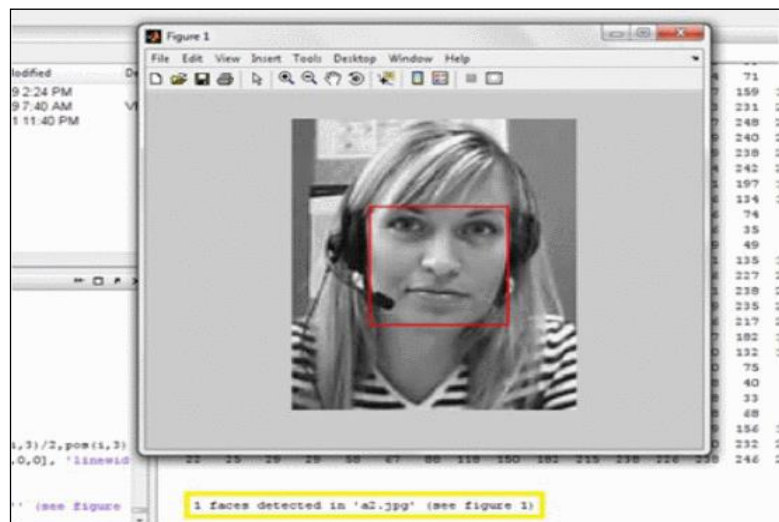
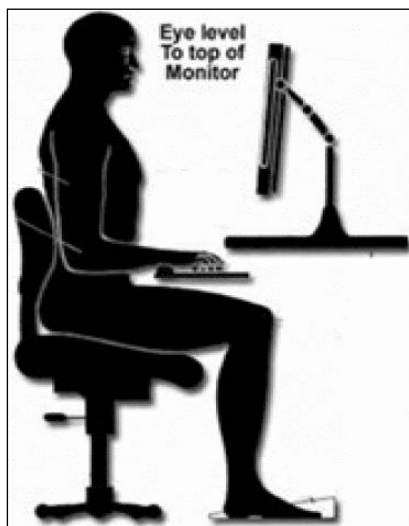


Fig 1: System setup for eye mouse

Fig 2: Image detection and image capture

Fig 3: Facial coordinate point



B. Eye Detection:

For eye detection, we use Eye-Aspect-Ratio (EAR). It was used to observe if the person's eye is flickering or not in the video frame. Each Eye is expressed as 6 coordinated (p1-p6), p1 is the coordinate of the left part of the eye and then p2-p6 is located accordingly when we travel in the clockwise direction as shown in (1).

$$EAR = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|}$$

p1-p6 are points located on the eye

p2-p6 and p3-p5 depicts the vertical distance between the eyes

p1-p4 depicts the horizontal distance between the eyes

eye-aspect-ratio (EAR) is consistent until the time eye is open, but it tends to zero when a flickering is started as shown in Fig 4.

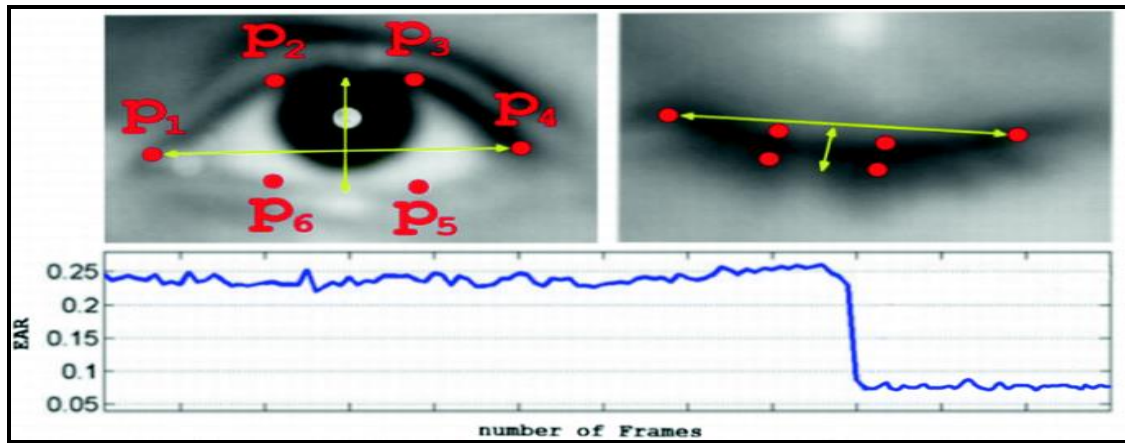


Fig 4: Top left: Eye is open. Top-right: Eye is closed. Bottom: Plotting the Eye-Aspect-Ratio (EAR) over time.

C. Mouth Detection:

Same as eye-aspect-ratio, we have mouth-aspect ratio (MAR), which is used to identify whether the mouth is open or not as shown in Fig 5

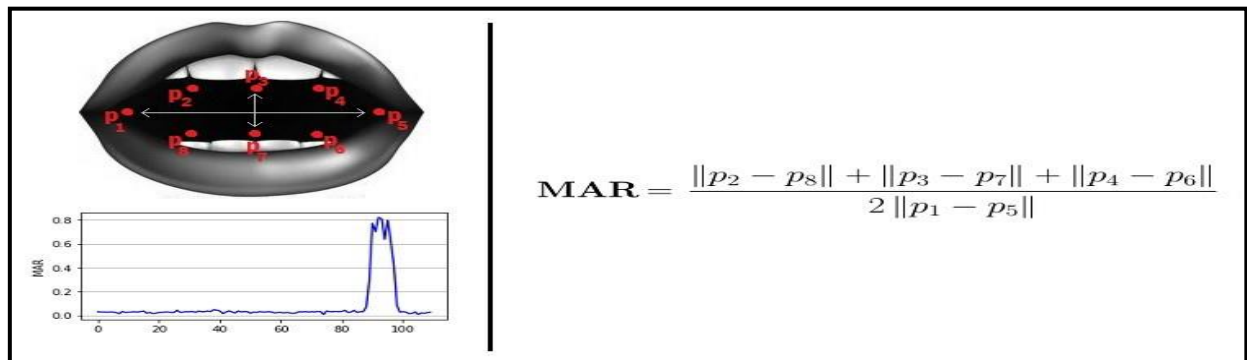


Fig 5: Mouth-Aspect-Ratio(MAR)

D. Usage:

The figure 8 explained the different modes and working of the algorithm. In this, we explained the action and their corresponding function we used in our algorithm.

- For activating/deactivate mouse control we have to open mouth continuously for some time.
- If the user blinks their left eye it will act as a left, click.
- If the user blinks their right eye it will act as a right-click.
- If the user squints his/her eye for some time it will activate/deactivate scrolling.
- Head will be used for cursor movements and scrolling up or down





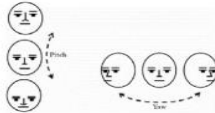
Action	Function
 Opening Mouth	Activate / Deactivate Mouse Control
 Right Eye Wink	Right Click
 Left Eye Wink	Left Click
 Squinting Eyes	Activate / Deactivate Scrolling
 Head Movements (Pitch and Yaw)	Scrolling / Cursor Movement

Fig 8: Action and it's corresponding function

CODE

```
import cv2
import dlib
import pyautogui
import mediapipe as mp
from scipy.spatial import distance
cam = cv2.VideoCapture(0)
face_mesh = mp.solutions.face_mesh.FaceMesh(refine_landmarks=True)
screen_w, screen_h = pyautogui.size()
flag = 0
def calculate_ear(eye):
    a = distance.euclidean(eye[0], eye[3])
    b = distance.euclidean(eye[1], eye[5])
    c = distance.euclidean(eye[2], eye[4])
    eye_aspect_ratio = (b+c)/(2*a)
    return eye_aspect_ratio

def calculate_mar(mou):
    a = distance.euclidean(mou[0], mou[4])
    b = distance.euclidean(mou[1], mou[7])
    c = distance.euclidean(mou[2], mou[6])
    d = distance.euclidean(mou[3], mou[5])
    mouth_aspect_ratio = (b + c + d) / (2 * a)
    return mouth_aspect_ratio

cap = cv2.VideoCapture(0)
hog_face_detector = dlib.get_frontal_face_detector()
dlib_facelandmark =
dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")
print("MODES:\n 0:sleep\n 1:Mouse pointer\n 2:Scroll\n")
print("Mode:", flag)
while True:
    _, frame = cap.read()
    frame = cv2.flip(frame, 1)
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    faces = hog_face_detector(gray)
    for face in faces:
        face_landmarks = dlib_facelandmark(gray, face)
        lefteye = []
        righteye = []
        mouth = []
        landmarks = []
        for n in range(36,42):
            x = face_landmarks.part(n).x
            y = face_landmarks.part(n).y
            lefteye.append((x,y))
            next_pt = n+1
            if n == 41:
                next_pt = 36
            x2 = face_landmarks.part(next_pt).x
            y2 = face_landmarks.part(next_pt).y
            cv2.line(frame, (x,y), (x2,y2), (0,255,0), 1)

        for n in range(42,48):
            x = face_landmarks.part(n).x
            y = face_landmarks.part(n).y
```

```

        landmarks.append(face_landmarks.part(n))
        righteye.append((x,y))
        next_pt = n+1
        if n == 47:
            next_pt = 42
        x2 = face_landmarks.part(next_pt).x
        y2 = face_landmarks.part(next_pt).y
        cv2.line(frame, (x,y), (x2,y2), (0,255,0), 1)

    for n in range(60, 68):
        x = face_landmarks.part(n).x
        y = face_landmarks.part(n).y
        mouth.append((x, y))
        next_pt = n+1
        if n == 67:
            next_pt = 60
        x2 = face_landmarks.part(next_pt).x
        y2 = face_landmarks.part(next_pt).y
        cv2.line(frame, (x,y), (x2,y2), (0,255,0), 1)

    left_ear = calculate_ear(lefteye)
    right_ear = calculate_ear(righteye)
    mouth_mar = calculate_mar(mouth)
    total_ear = (left_ear + right_ear)/2
    rgb_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    output = face_mesh.process(rgb_frame)
    landmark_points = output.multi_face_landmarks
    frame_h, frame_w, _ = frame.shape

    if flag==0:
        cv2.putText(frame, "Select a Mode", (20,
100),cv2.FONT_HERSHEY_PLAIN, 3, (0, 0, 255), 4)
    elif flag==1:
        cv2.putText(frame, "Mouse Pointer Activated", (20,
100),cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 4)
    elif flag==2:
        cv2.putText(frame, "Scrolling Activated", (20,
100),cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0), 4)

    if mouth_mar > 0.5:
        if flag == 0:
            flag=1
            print("Mode:", flag)
        elif flag == 1:
            flag = 0
            print("Mode:", flag)
        else:
            continue
        pyautogui.sleep(1)
    if total_ear<0.15:
        if flag == 0:
            flag=2
            print("Mode:", flag)
        elif flag == 2:
            flag = 0
            print("Mode:", flag)
        else:
            continue
        pyautogui.sleep(1)
    if flag == 1:
        if landmark_points:

```

```

        landmarks = landmark_points[0].landmark
        for id, landmark in enumerate(landmarks[474:478]):
            x = int(landmark.x * frame_w)
            y = int(landmark.y * frame_h)
            cv2.circle(frame, (x, y), 3, (0, 255, 0))
            if id == 1:
                screen_x = screen_w * landmark.x
                screen_y = screen_h * landmark.y
                pyautogui.moveTo(screen_x, screen_y)
                if right_ear < 0.2:
                    pyautogui.rightClick()
                    pyautogui.sleep(1)
                if left_ear < 0.2:
                    pyautogui.click()
                    pyautogui.sleep(1)

        if flag == 2:
            m = 265-face_landmarks.part(33).y
            pyautogui.scroll(m)

    cv2.imshow("mouse control", frame)
    key = cv2.waitKey(1)
    if key == 27:
        break
cap.release()
cv2.destroyAllWindows()

```

RESULT

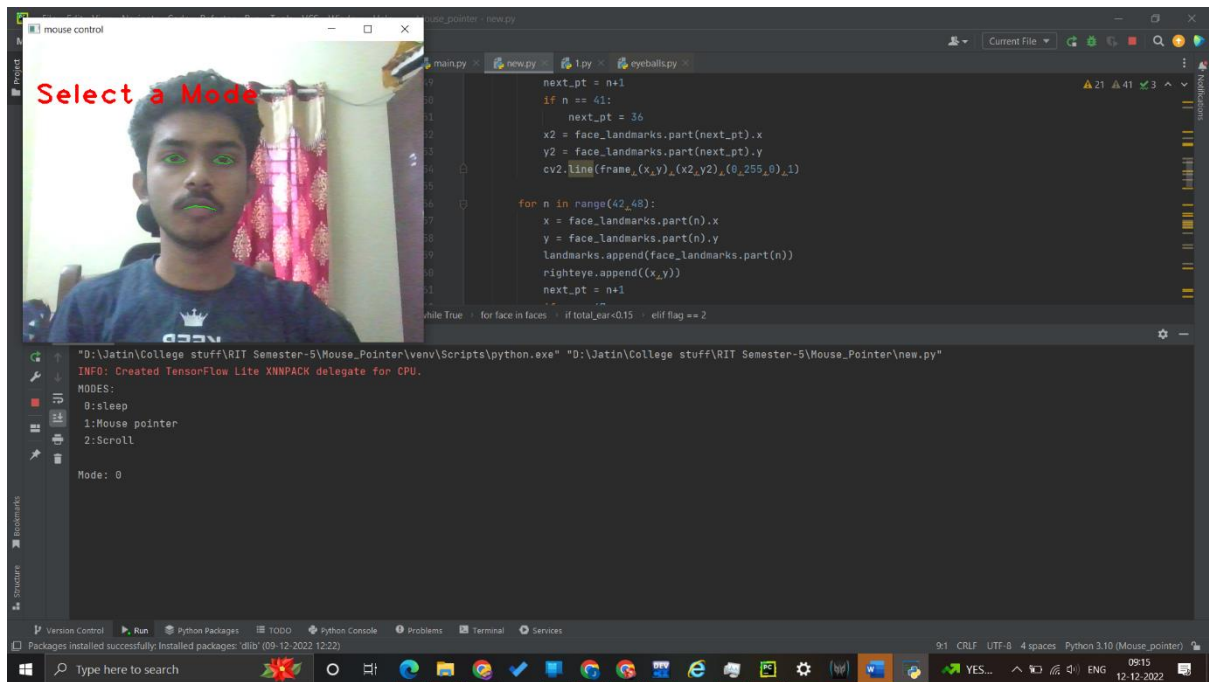


Fig 9: Mode Selection

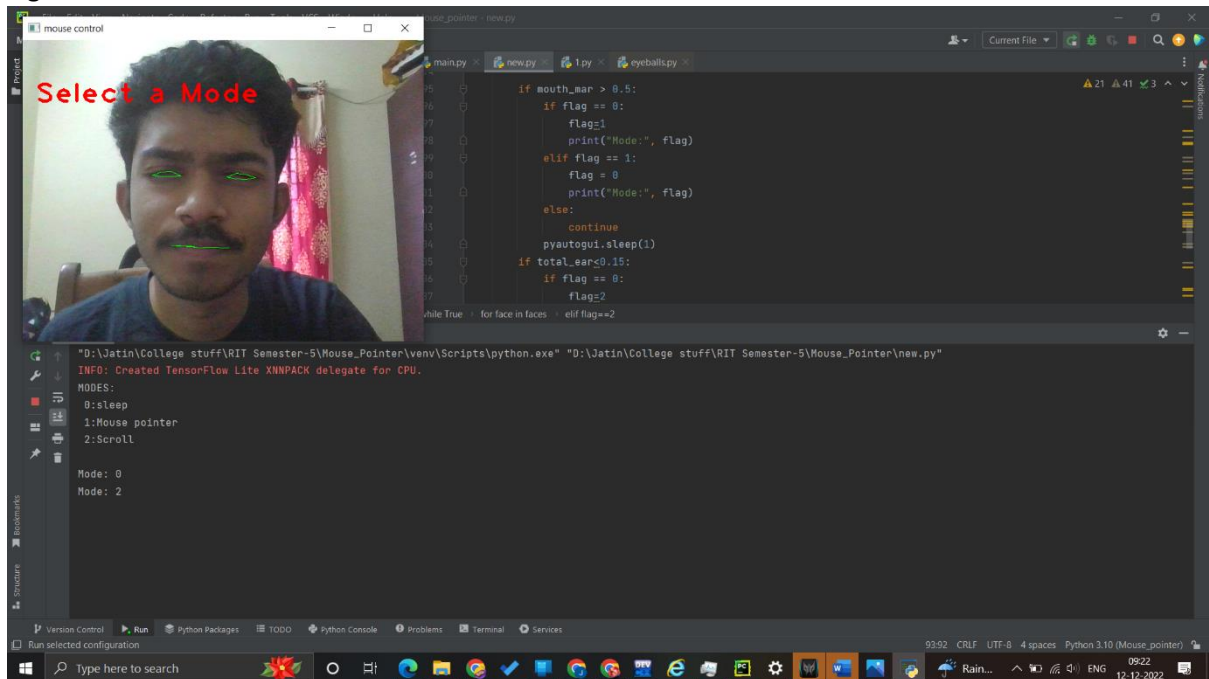


Fig 10: Squinting eyes for activating scroll mode

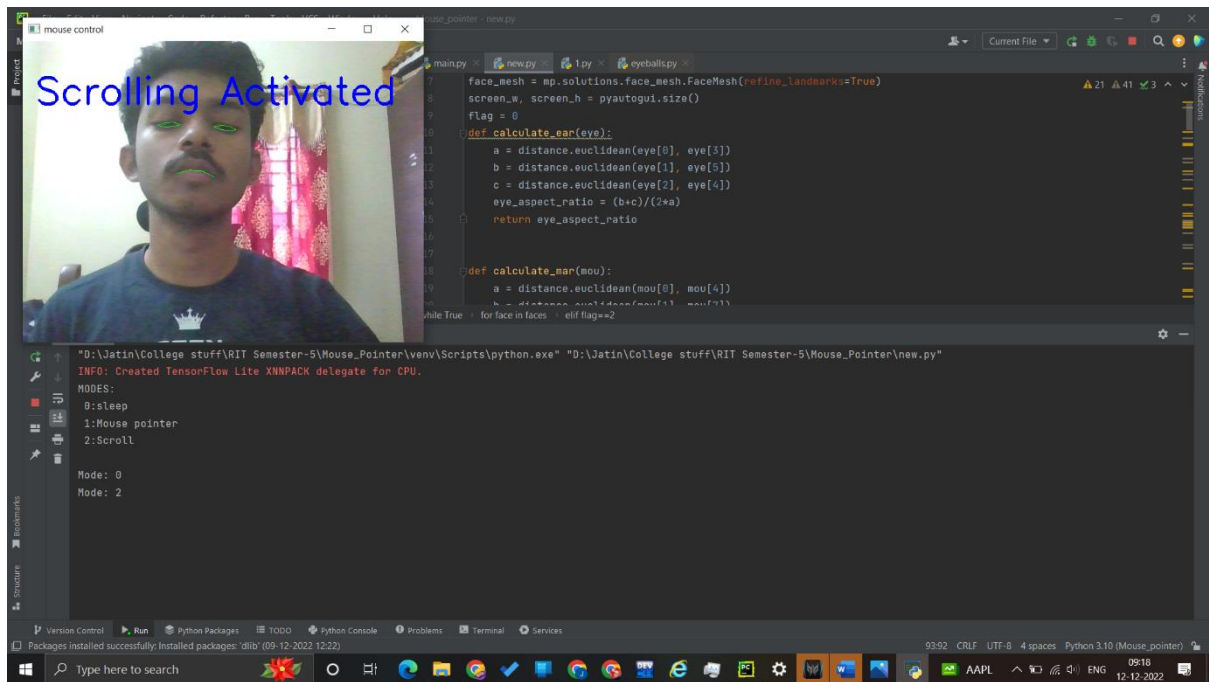


Fig 11: Scrolling Mode Activated

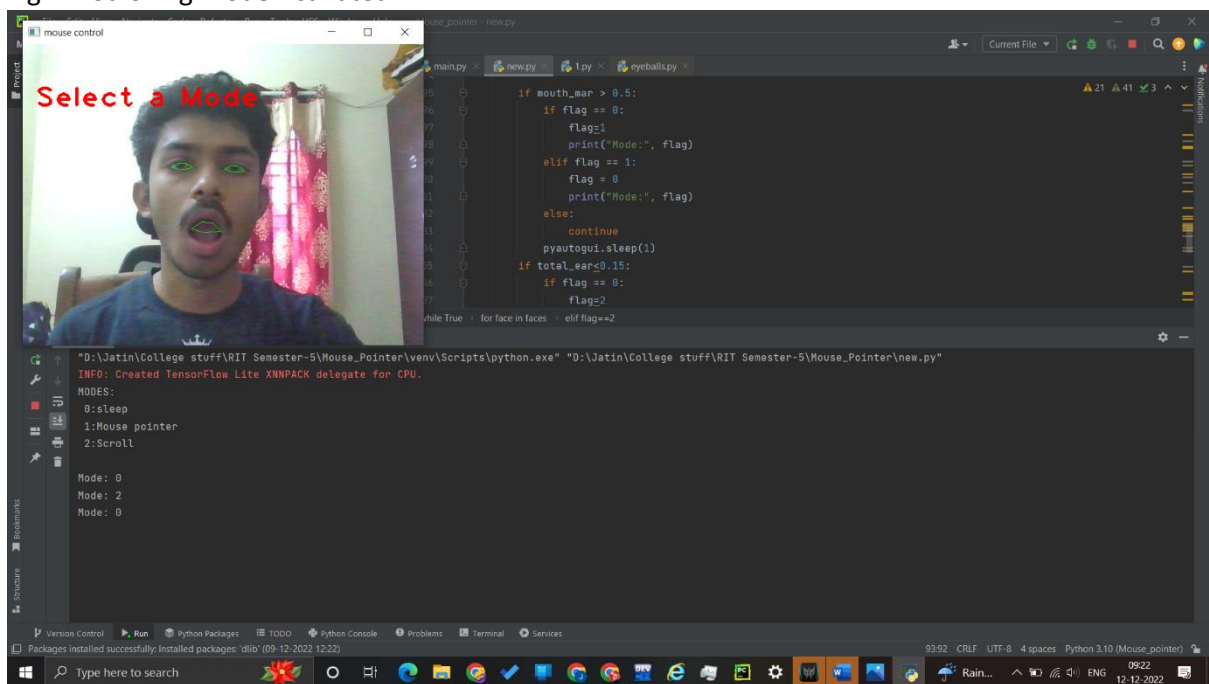


Fig 12: Opening mouth for activating Mouse

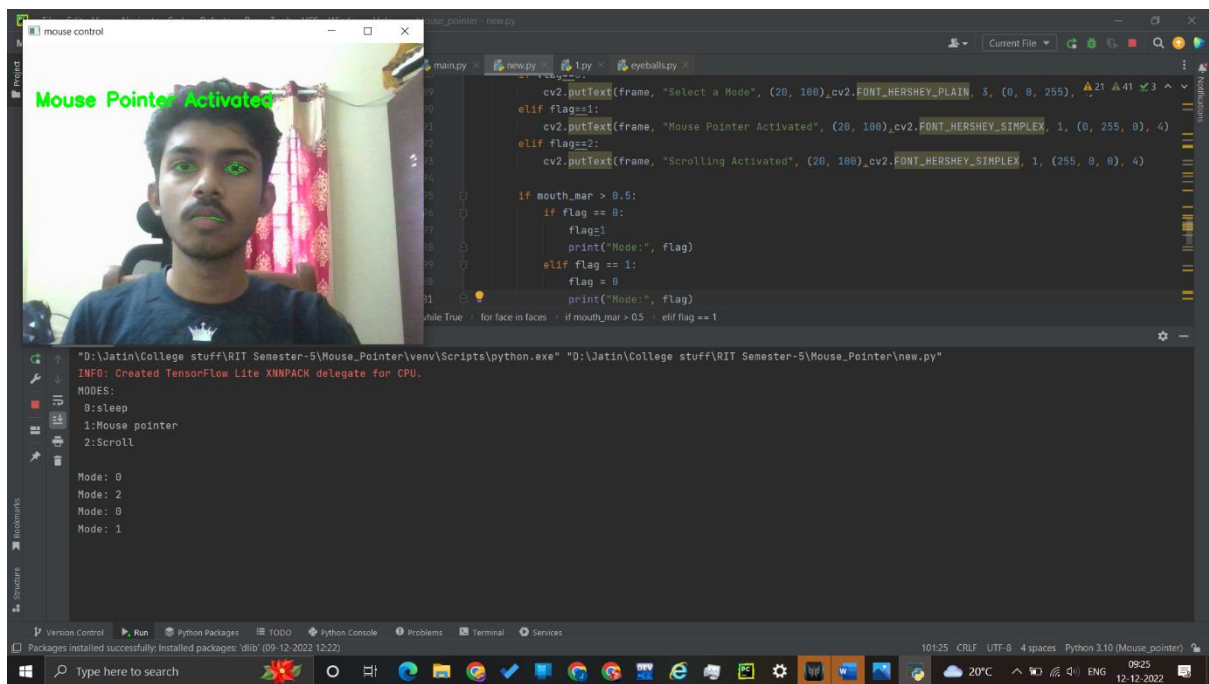


Fig 13: Moving Face for Moving Pointer

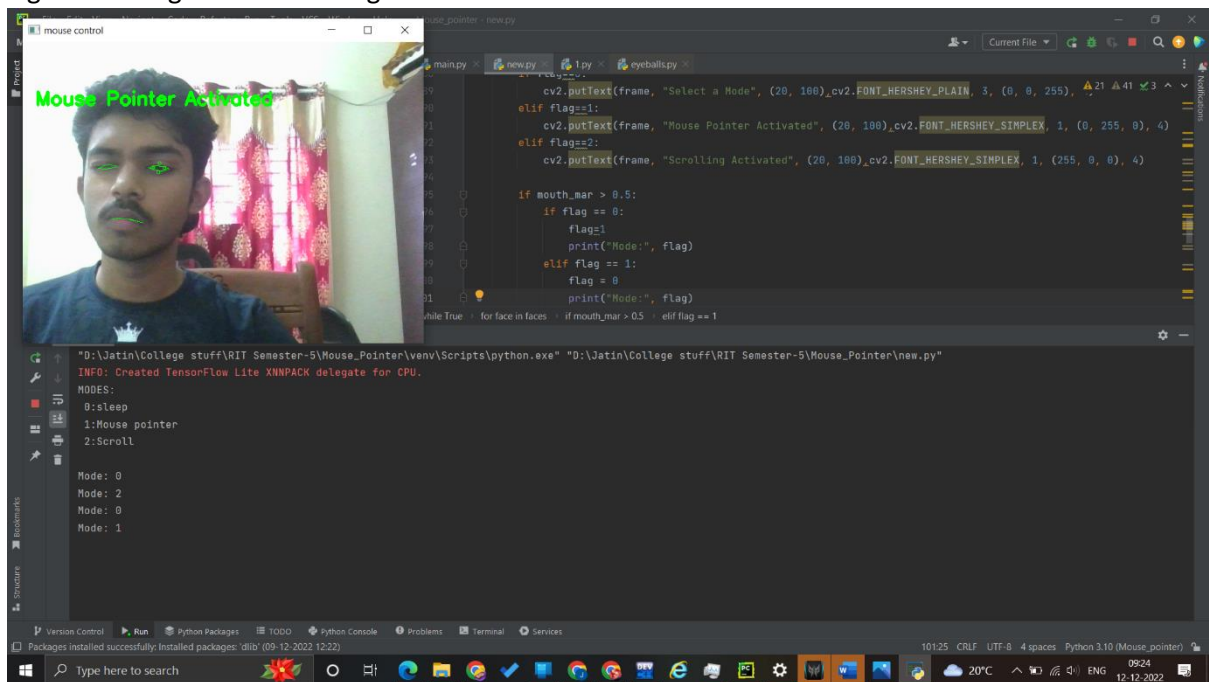


Fig 14: Left wink for left click

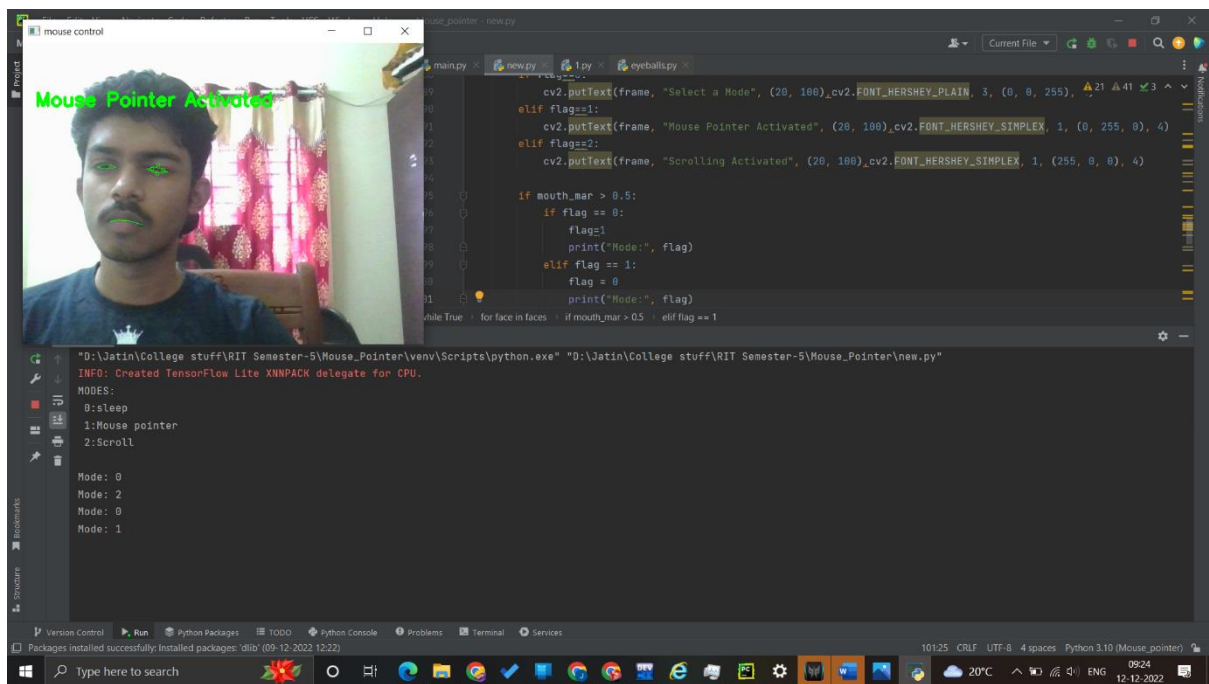


Fig 15: Right wink for Right click

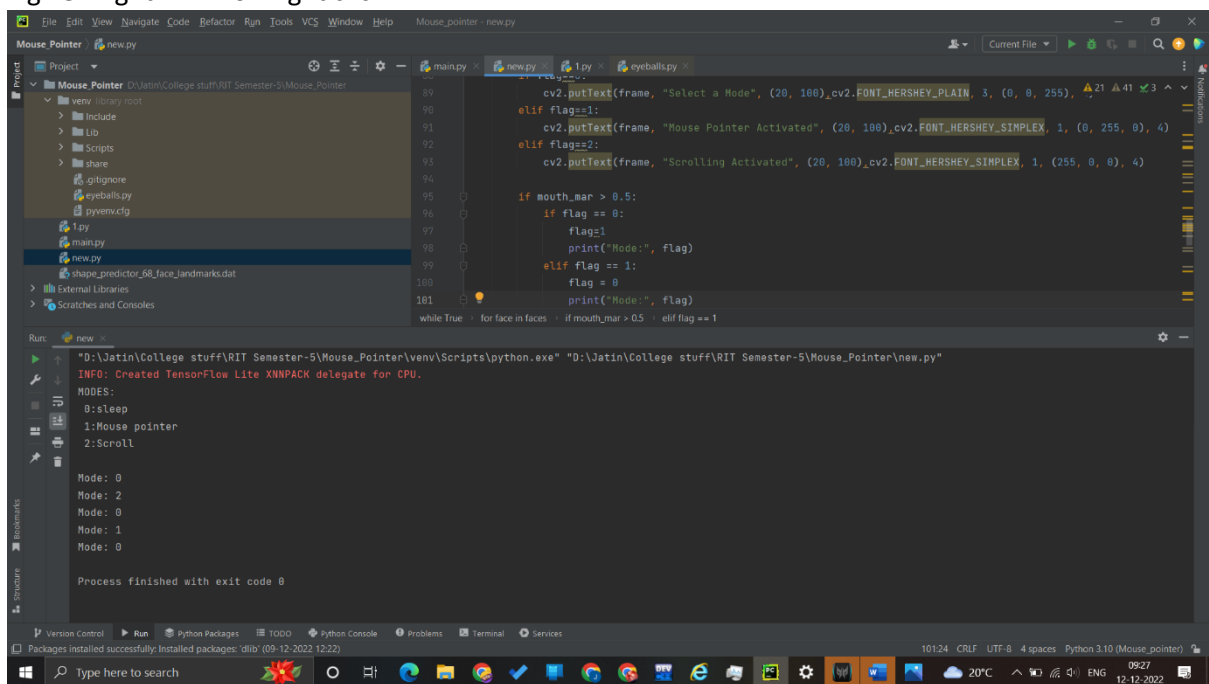


Fig 16: Press esc for exiting

Project schedule

Sl No	Process	Date
1	Topic Selection	20-10-2022
2	Idea Presentation	03-11-2022
3	Code Implementation	01-12-2022
4	Report Making	10-12-2022
5	Report Submission	12-12-2022

References

1. <https://ibug.doc.ic.ac.uk/resources/300-W/>
2. <https://pyautogui.readthedocs.io/en/latest/mouse.html>
3. http://dlib.net/face_landmark_detection.py.html