FACIAL MOUSE CONTROLLING SYSTEM

A project report submitted to

Rajiv Gandhi University of Knowledge Technologies

SRIKAKULAM

In partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

3rd year B. Tech 2nd semester

Submitted by
G. Syam venkata sai (S190055)
V. R.S.S.Durga (S180732)
K. pavani (S190422)
D. Lokesh (S190127)

Under the Esteemed Guidance of

Mrs. Ch. Lakshmi Bala M.Tech, (Ph.D)

Assistant Professor

Department of Computer Science and Engineering,
RGUKT-SRIKAKULAM, ETCHERLA.

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

I. DECLARATION

We declared that this thesis work entitled "Facial mouse controlling System" is carried out by us during the year 2023-24 in partial fulfillment of the requirements for the Minor Project in Computer Science and Engineering.

We further declare that the work contained in this report is original and has been done by us under the guidance of my supervisor(s). The work has not been submitted to any other Institute for any degree or diploma. We have followed the guidelines provided by the University in preparing the report. We have confirmed the norms and guidelines given in the Ethical Code of Conduct of the University. Furthermore, the technical details furnished in various chapters of this thesis are purely relevant to the above project and there is no deviation from the theoretical point of view for design, development and implementation.

Date:

Place: Srikakulam

G. Syam venkata sai (S190055)
V. R.S.S.Durga (S180732)
K. pavani (S190422)
D. Lokesh (S190127)

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

II. BONAFIDE CERTIFICATE

Certified that this project work titled “**FACIAL MOUSE CONTROLLING SYSTEM**” is the bonafide work of **Mr. D Lokesh (S190127)** , **Ms. K Pavani (S190422)** , **Ms. V.R.S.S.Durga (S180732)** and **Mr. G Syam venkata sai (S190055)”** who carried out the work under my supervision, and submitted in partial fulfillment of the requirements for the award of the degree, BACHELOR OF TECHNOLOGY, during the year 2023 - 2024.

Mrs. Ch. Lakshmi Bala M.Tech, (Ph.D)
Assistant Professor,
Project Guide,
Department of CSE,
RGUKT-SKLM

Mrs. Ch. Lakshmi Bala M.Tech, (Ph.D)
Assistant Professor,
Head of the Department,
Department of CSE,
RGUKT- SKLM

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

III. ABSTRACT

As webcams and other affordable devices like sensors are readily accessible on the market, the Human Computer Interaction (HCI) is used in a variety of ways. Gestures are the most effective means of communication between humans and machines. In recent years, new approaches to HCI have been created, such as interacting with a machine using touch, voice, facial expressions, and head, hand and head motions. The goal of HCI is to make computers more user-friendly in order to enhance interactions between humans and computers.

"Facial Mouse Controlling System" aims to let users control a virtual mouse with facial expressions. In the current world, interacting with computers is essential. Physically disabled people may find it challenging to use a mouse, thus we have suggested this mouse control utilizing facial expressions as a possible option for those individuals. They can move the mouse with their face by using their eyes, nose and mouth. The user has the choice to blink, which causes the mouse to be clicked when using the left eye and while using the right eye. Squinting your eyes will start the scroll mode. The pointer moves in one of two directions: left or right.

Keywords: python, Computer vision, Facial gestures, dlib

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

IV. ACKNOWLEDGEMENT

We want to express our sincere gratitude and debt of gratitude to our project's mentor, Assistant Professor Mrs. Ch. Lakshmi Bala madam, who served as a continual source of inspiration and direction during the project's duration. Being able to work under her direction and successfully finish the thesis work has been a wonderful pleasure for us.

We wish to extend our sincere thanks to Mrs. Ch. Lakshmi Bala madam, Assistant professor, Head of the Computer Science and Engineering Department, for her constant encouragement throughout the project.

We extend our gratitude to everyone who assisted us, whether directly or through other means, in finishing our project work.

Project Associate:

G. Syam venkata sai (S190055)
V. R.S.S.Durga (S180732)
K. pavani (S190422)
D. Lokesh (S190127)

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER-1

INTRODUCTION

1.1. Introduction:

The facial mouse control system is an innovative technology that enables individuals with physical disabilities to interact with computers and other digital devices using only their facial movements. This system is designed to provide a reliable and efficient means of communication and control for users who may not be able to use traditional input devices such as a keyboard or a mouse. The aim of this system is to provide an easy-to-use and intuitive interface that can be accessed by anyone, regardless of their physical limitations. With this system, users can control the movement of the mouse pointer on a computer screen and perform a variety of actions such as clicking, scrolling, and typing using just their facial expressions.

1.2. Problem Statement:

The goal of this project is to create a facial mouse-controlling system that will allow people with limited or no physical ability to interface with and operate computers just by using their facial expressions. The existing approaches of computer interaction mainly rely on manual input tools like a mouse or keyboard, which present major challenges for people.

Therefore, there is a need for a non-intrusive, effective, and precise system that can monitor and determine facial expressions to control the mouse cursor and carry out various tasks on a computer, giving people with disabilities an efficient means of computer interaction and improving their overall quality of life.

1.3. Objectives:

- The project aims to provide a solution for physically disabled individuals, particularly those without hands and cannot speak may find difficult to use a traditional computer mouse
- By utilizing facial gestures, such as raising eyebrows or opening the mouth, users can control the movement of the mouse pointer on the screen.
- This technology has the potential to make interacting with computers and other digital devices more accessible and easier for individuals who have suffered a physical injury affecting their ability to use their hands.
- This virtual mouse has the potential to make interacting with computers and other digital devices more accessible and easier for individuals who have suffered a physical injury affecting their ability to use their hands.
- Facial Mouse control system can enable users to perform various tasks, including clicking on icons, scrolling through documents, or typing using an onscreen keyboard.

1.4. Goals:

- To make physically challenged people's computer interactions more effective and efficient, so boosting their capacity for work and study in a digital world.
- To provide physically challenged individuals with an alternative and more accessible way of interacting with computers, particularly when taking online exams.
- To lessen the use of traditional input devices like a keyboard and mouse, which can be difficult for people with physical disabilities to operate.

1.5. Scope:

- The primary scope of a facial mouse controlling system is to provide a means of computer access for individuals with limited mobility. It can empower people with conditions such as spinal cord injuries, muscular dystrophy, cerebral palsy, or paralysis to independently interact with computers, tablets, or smartphones.
- Facial mouse control can serve as an alternative input method for anyone who prefers or requires hands-free computer interaction.
- Implementing different facial gestures to perform various actions, such as clicking, scrolling.

1.6. Applications:

- **Accessibility for Motor Impairments:**

  Facial mouse systems can enable individuals with motor impairments, such as paralysis or limited mobility in their limbs, to control a computer mouse solely using their facial movements.
- **Virtual Reality (VR) and Gaming:**

  By mapping facial gestures to in-game actions, users can interact with virtual environments or play games using intuitive facial movements.
- **Personal Computing:**

  Facial mouse control can be used by anyone looking for alternative input methods or seeking hands-free operation of their computer. It can be particularly useful in situations where manual mouse use is inconvenient or not possible, such as when hands are occupied, during certain medical procedures, or for individuals with temporary injuries or conditions.
- **Communication:**

  This can be used to develop communication devices that improve the capacity of people with physical limitations to interact and communicate with others by allowing them to express themselves through facial expressions.

1.7. Limitations:

- Facial mouse control relies on facial tracking and gesture recognition technology. However, the accuracy and precision of these systems may vary. Factors like lighting conditions, camera quality, and individual facial features can affect the system's ability to accurately track and interpret facial movements. This can result in occasional inaccuracies or delays in cursor movement, which may impact user experience and efficiency.
- Facial mouse controlling systems often require specific hardware components, such as cameras or sensors capable of facial tracking. Not all devices may have built-in support for such technology, and additional hardware or peripherals may be necessary.

CHAPTER-2

LITERATURE SURVEY

2.1. Collect Information:

The development of techniques for analyzing eye movements has gained significant interest in recent years. This literature survey explores two research papers that contribute to the understanding and implementation of a facial mouse controlling system, focusing on eye-blink detection and fatigue detection using facial landmarks.

2.2. Study:

Eye-blink detection system for human-computer interaction:

Aleksandra Krolak and Paweł Strumiłło presented a vision-based human-computer interface in their paper published in 2011. The interface recognizes and interprets voluntary eye blinks as control commands. Haar-like features for automatic face detection, template matching-based eye tracking, and eye-blink detection are among the image processing approaches used.

The interface was tested on 49 users, including 12 with physical disabilities. The test results demonstrated the interface's usefulness in offering an alternative means of communication with computers. Users were able to enter English and Polish text with an average time of less than 12 seconds per character and browse the Internet. The interface is based on a notebook equipped with a typical web camera and requires no additional light sources. The availability of the interface as open-source software allows for further exploration and adaptation for a facial mouse controlling system.

Real-time fatigue detection using shape predictor 68 face landmarks algorithm:

The use of Dlib's facial landmark detector and the methodology of detecting and monitoring important facial landmarks for fatigue detection is discussed in this paper. Face detection is the initial step, locating a human face and returning the face's position as a rectangle in coordinate form. Subsequently, facial landmarks are determined within that rectangle. The paper introduces the use of a pre-built model in Dlib for face detection and demonstrates the calculation of Euclidean distance between the upper and lower eyelids for drowsiness detection. This method is also suitable for individuals wearing glasses. The integration of this fatigue detection technique using facial landmarks can enhance the functionality of a facial mouse controlling system by detecting user fatigue and adapting the system accordingly.

In conclusion, the literature survey highlights two important research papers that contribute to the development of a facial mouse controlling system. By incorporating these findings into the design and implementation of a facial mouse controlling system, it is possible to enhance the interaction between users and computers, particularly for individuals with physical disabilities or those experiencing fatigue during computer use.

2.3. Benefits:

- **Independence and Empowerment:**

  Facial mouse controlling systems empower users by giving them control over their digital environment. Individuals with physical disabilities can access and use computers without relying on assistance from others, promoting independence and boosting self-confidence.
- **Hands-Free Interaction:**

  It enables hands-free interaction with computers, allowing users to perform tasks without requiring the ability to use their hands. This is particularly beneficial for individuals with conditions such as paralysis, limb differences, or temporary injuries that affect hand movements.
- **Continuous Advancements:**

  Facial recognition and computer vision technologies continue to advance, leading to improvements in accuracy, responsiveness, and usability of facial mouse controlling systems. Ongoing research and development aim to enhance the performance and expand the capabilities of these systems.

2.4. Hardware Components:

- **Computer or Digital device:**

  A computer or digital device capable of running the facial mouse controlling system.
- **Webcam:**

  - A high-quality webcam capable of capturing facial expressions and movements accurately.
  - A minimum resolution of 2mp or higher is recommended to capture detailed facial features.
- **Connectivity:**

  The computer or digital device should have the necessary ports or wireless connectivity options to connect the webcam, if applicable.
- **Minimum 4GB of RAM:**

  The facial mouse controlling system needs to be able to identify the user's face from a variety of angles and lighting conditions. This requires a lot of data to be stored and processed. So, at minimum it requires 4GB Ram.
- **Minimum 1GB of storage:**

  A minimum of 1 GB of storage is required for a facial mouse controlling system because it needs to store a variety of data including Facial templates, Software settings.
- **AMD processor or higher with any generation:**
- **500 GB Hard disk or above**

2.5. Software Components:

- Windows 10 or above versions
- Python 3.10.5 or above versions or Anaconda

2.6. Technologies used:

- Opencv
- Numpy
- Dlib
- PyautoGUI
- Imutils

2.7. Summary:

- A facial mouse controlling system is a software solution designed to enable users to control the computer mouse using facial movements and gestures.
- It typically involves the integration of various software components, such as facial recognition, gesture recognition, mouse emulation, calibration and configuration, user interface, accessibility integration, compatibility and integration components.
- The system captures facial features and movements through a camera or sensor, interprets them into specific commands or actions, and emulates the behaviour of a traditional mouse.
- Users can customize settings, calibrate the system, and interact with the system through a graphical user interface.
- Facial mouse controlling systems are often used as assistive technology for individuals with disabilities, providing alternative input methods and enhancing accessibility.

CHAPTER-3

SYSTEM ANALYSIS

3.1. Existing model:

- We have an existed system of Computer cursor control using eye.
- We have an existed system of Computer cursor control using hands.
- We have an existed system of Facial gesture interfaces for expression and communication.

3.2. Disadvantages:

- Facial gestures are not as precise as mouse movements, so controlling a mouse with facial gestures may not be as accurate as using a traditional mouse.
- The existed system doesn't have zoom in and zoom out features.
- The existed system doesn't work properly, when the users use their spectacles.
- Low lightening conditions, the users face is not clearly detected under low light .The system can only operate on devices with an integrated webcam.

3.3. Proposed Model:

- Our proposed system is a cursor control system that uses facial expressions as input, providing a new and innovative way to control cursor movements on a computer screen.
- **Using Facial Gestures:**

  Our system has several features that make it unique and superior to existing cursor control systems
- **Using Spectacles:**

  Our model enables users to operate the system while wearing special equipment called Spectacles.
- **Responsiveness:**

  Our system is designed to be responsive and accurate like traditional input devices such as a mouse or trackpad. This allows for faster and more intuitive control of the cursor.
- **Accuracy:**

  Our system is designed to accurately interpret facial expressions, ensuring that users can control the cursor with precision and accuracy.

Overall, our proposed system is an innovative solution that addresses the problem of accessibility and ease of use for users who may have difficulty using traditional input devices, as well as providing an alternative input method that can be faster and more intuitive for all users.

3.4. Advantages:

- **Accessibility:**

  Facial gesture-based control can provide an accessible interface for individuals with physical disabilities or limited mobility, allowing them to control a computer mouse with facial movements.

- **Convenience:**

  Users can control the mouse without having to physically touch it, providing a convenient and intuitive interface for controlling the computer.

- **Precision:**

  Facial gestures can provide a high level of precision for controlling the mouse cursor, allowing users to navigate and interact with the computer with greater accuracy.

- **Efficiency:**

  With facial gesture-based control, users can perform mouse operations quickly and easily, without having to rely on a physical mouse or keyboard.

- **Reduced risk of injury:**

  Using facial gestures to control the mouse can help reduce the risk of repetitive strain injuries and other physical ailments associated with prolonged mouse use.

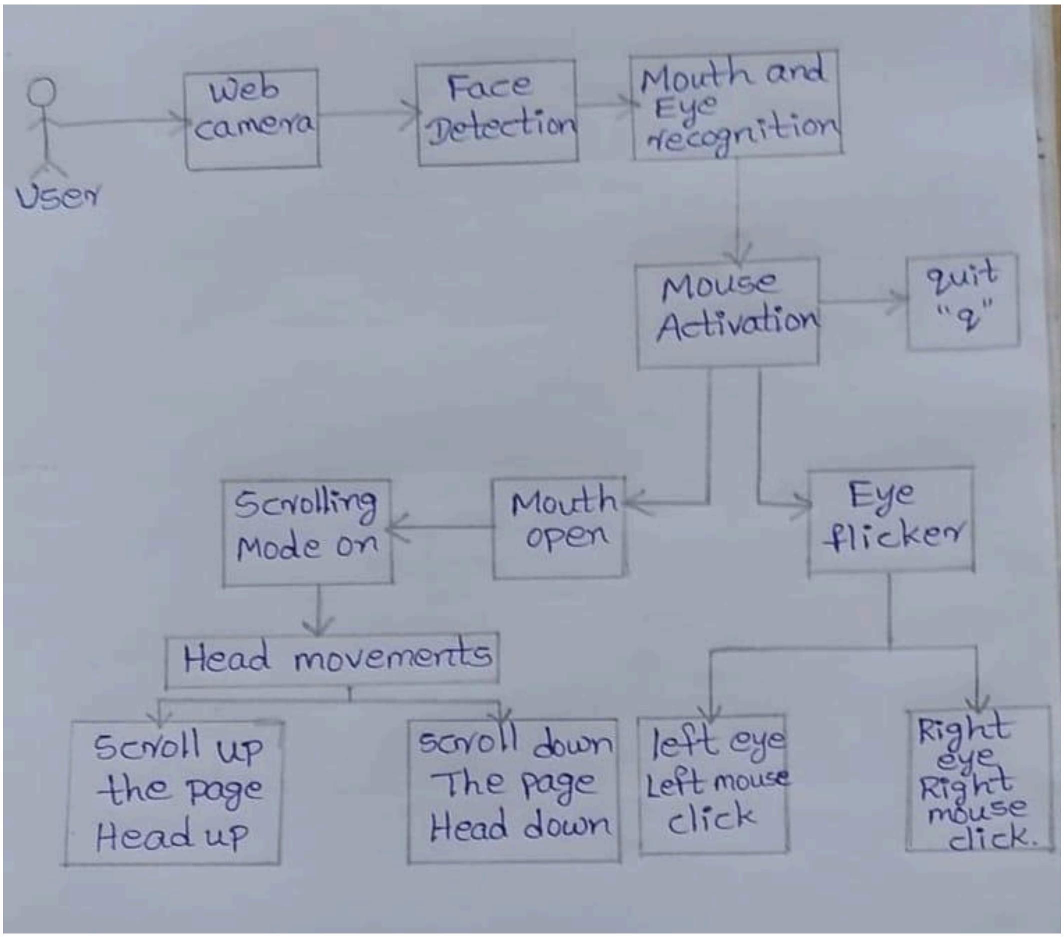
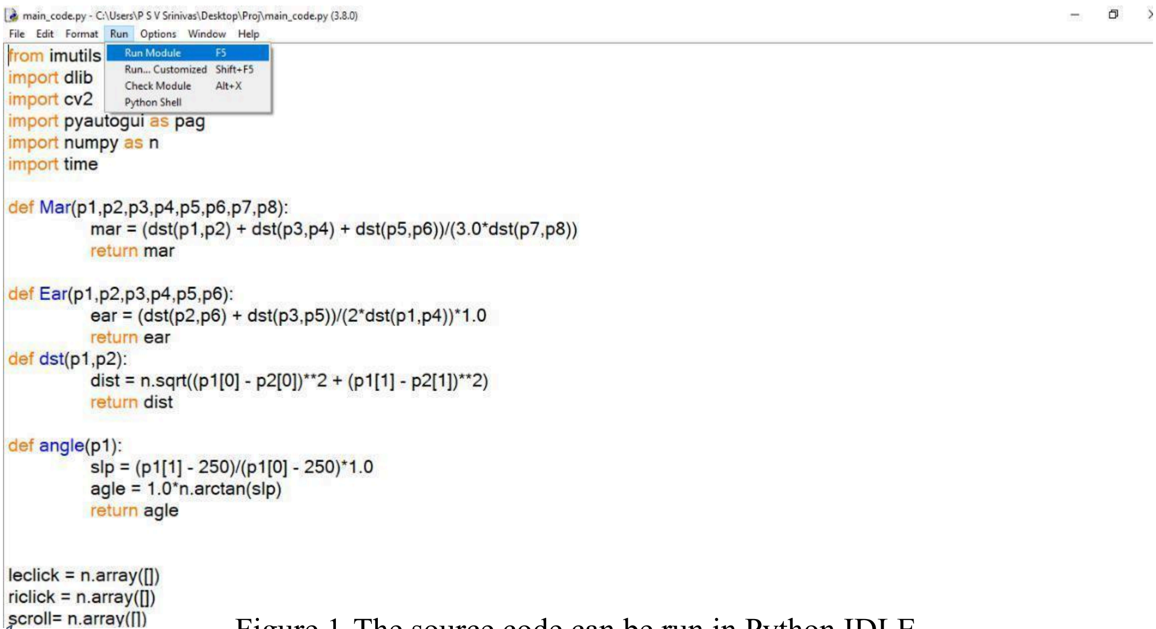


Fig: Proposed system WorkFlow

CHAPTER-4

SYSTEM IMPLEMENTATION




```
from imutils
import dlib
import cv2
import pyautogui as pag
import numpy as n
import time

def Mar(p1,p2,p3,p4,p5,p6,p7,p8):
            mar = (dst(p1,p2) + dst(p3,p4) + dst(p5,p6))/(3.0*dst(p7,p8))
            return mar

def Ear(p1,p2,p3,p4,p5,p6):
            ear = (dst(p2,p6) + dst(p3,p5))/(2*dst(p1,p4))*1.0
            return ear
def dst(p1,p2):
            dist = n.sqrt((p1[0] - p2[0])**2 + (p1[1] - p2[1])**2)
            return dist

def angle(p1):
            slp = (p1[1] - 250)/(p1[0] - 250)*1.0
            agle = 1.0*n.arctan(slp)
            return agle


leclick = n.array([])
riclick = n.array([])
scroll= n.array([])
```

1 Figure 1-The source code can be run in Python IDLE

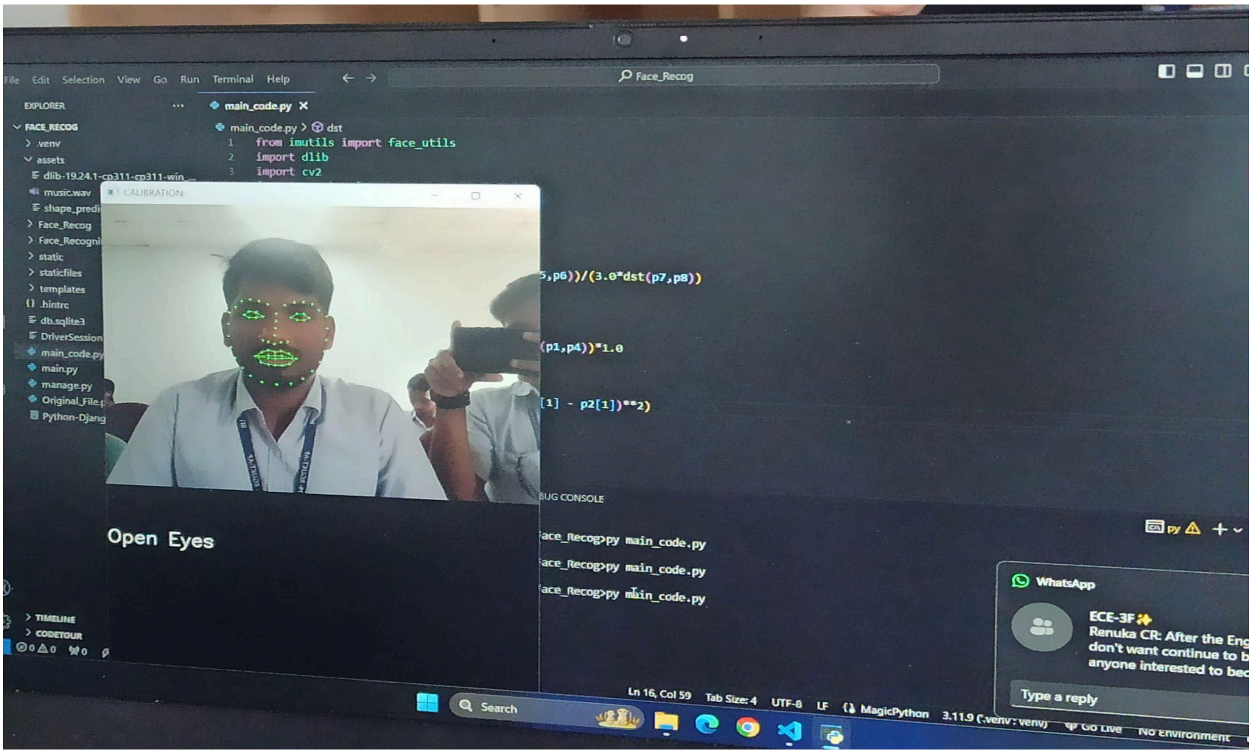


Figure 2 Taking input when both eyes opened

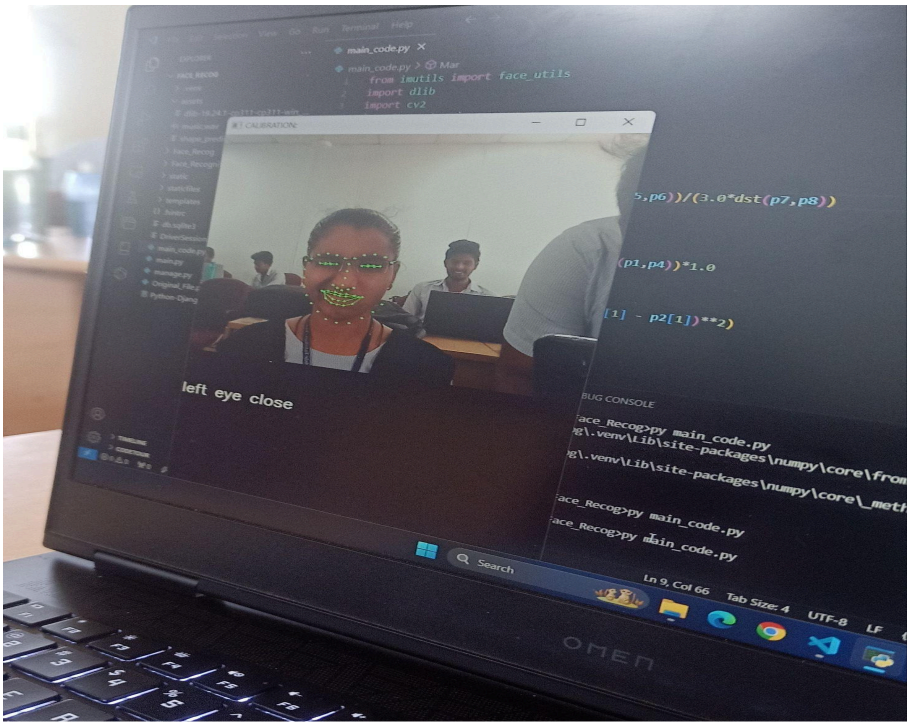


Figure 3 Taking input while closing left eye

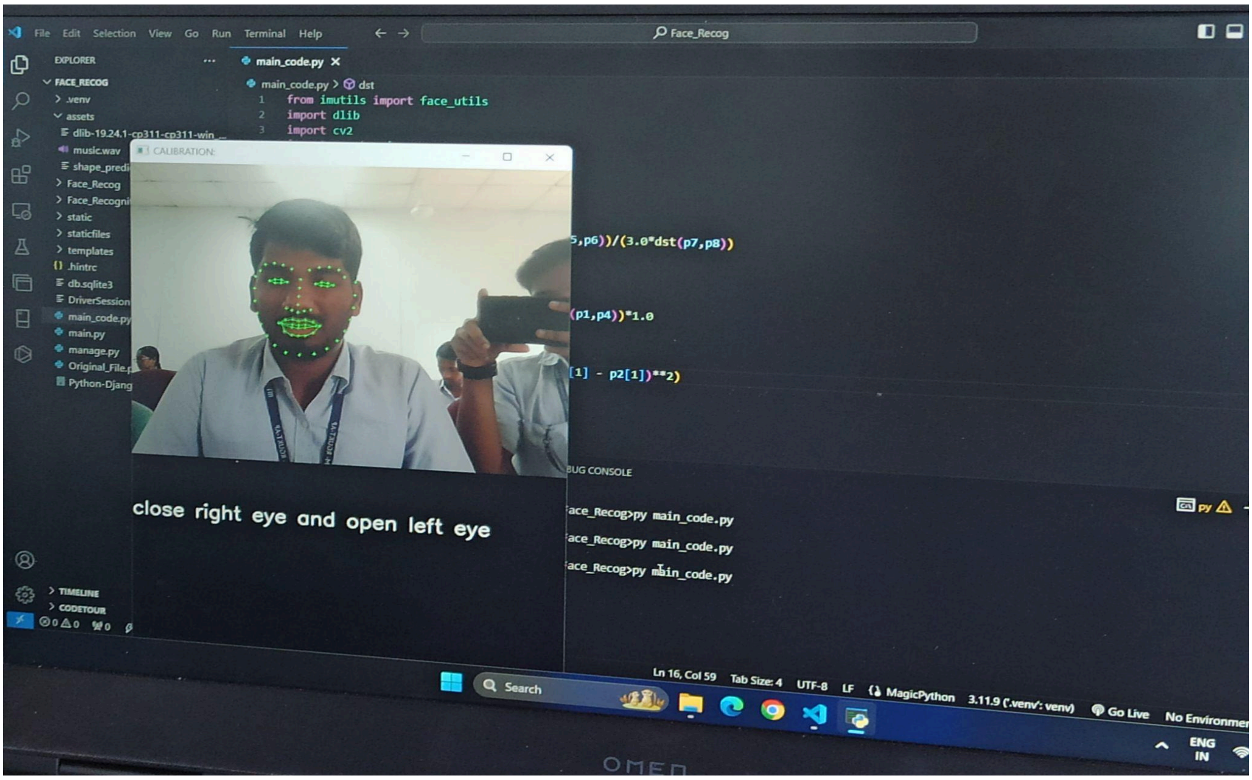


Figure 4 - Taking input while closing right eye

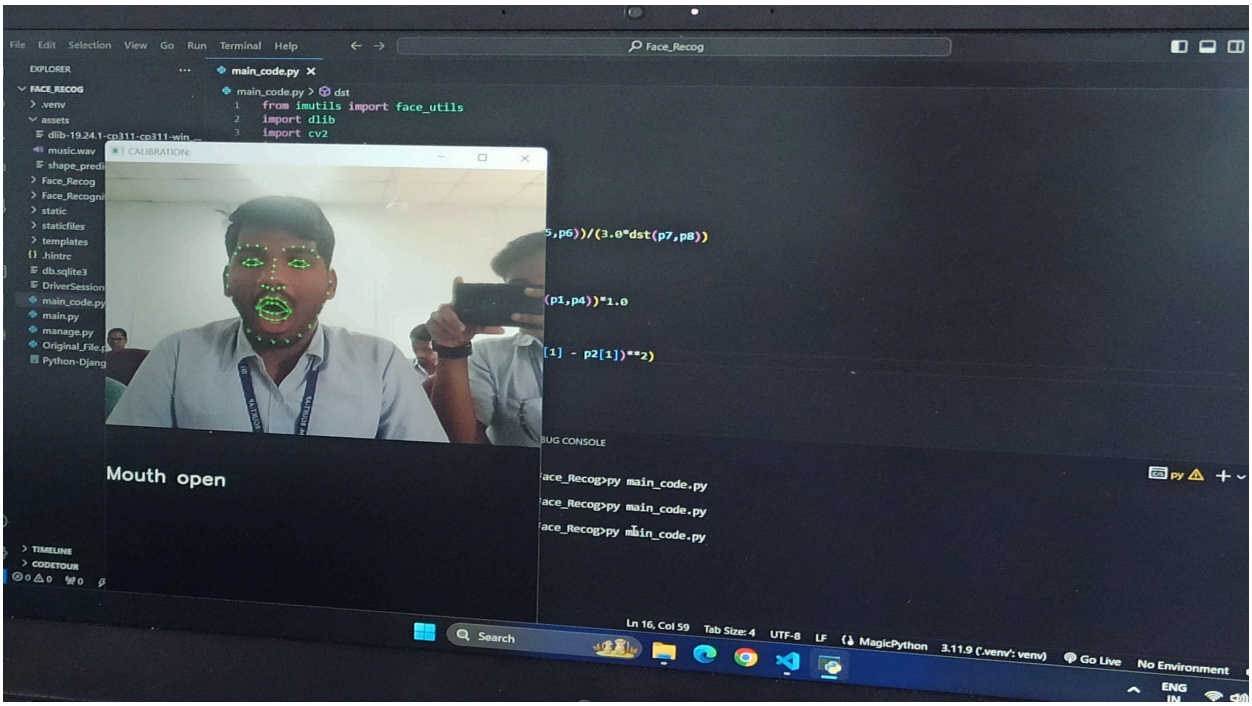


Figure 5-Taking input by opening mouth

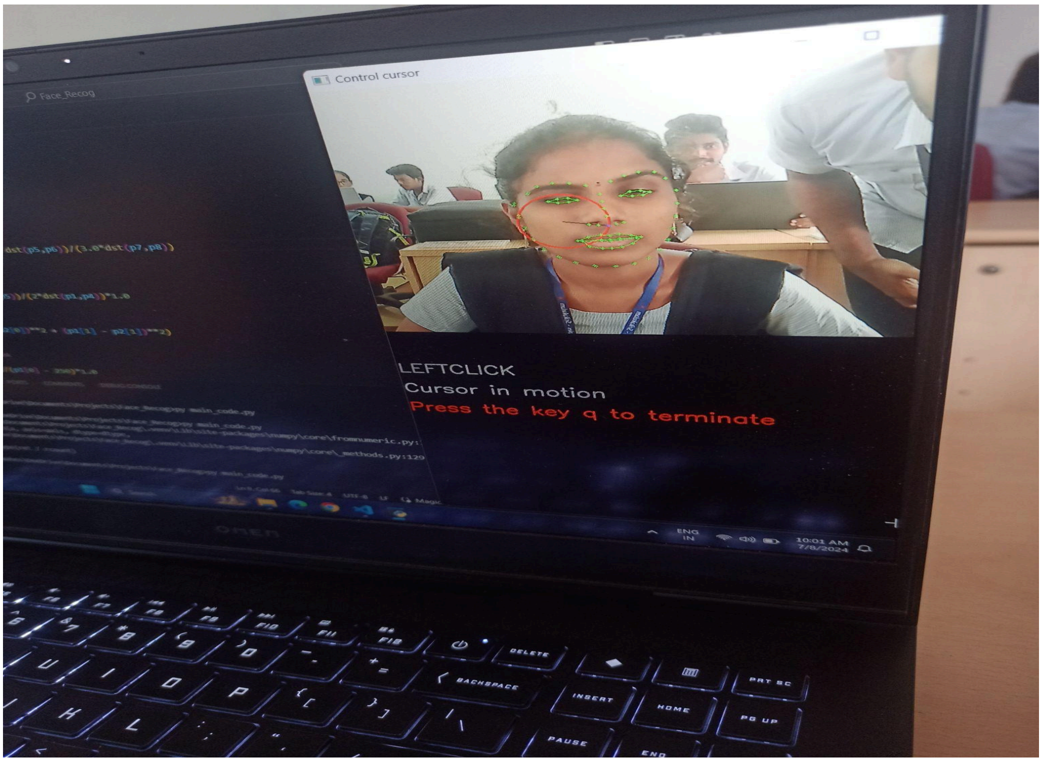


Figure 6 - Winking the left eye can achieve a left mouse click operation

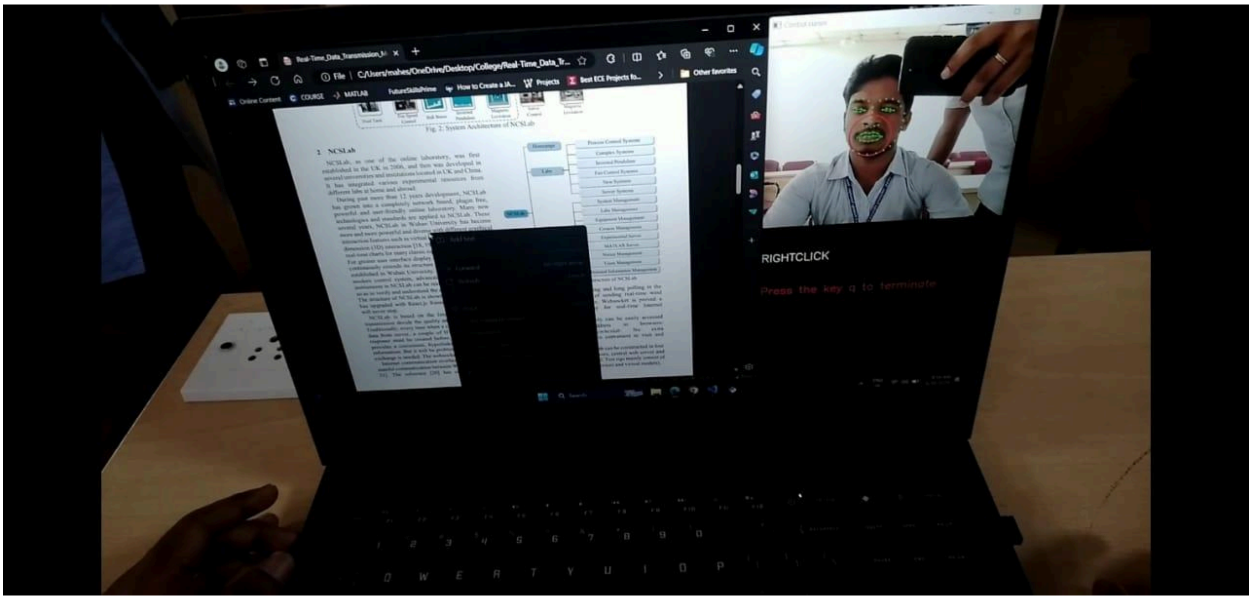


Figure 7 - Winking the right eye can achieve a right mouse click operation.

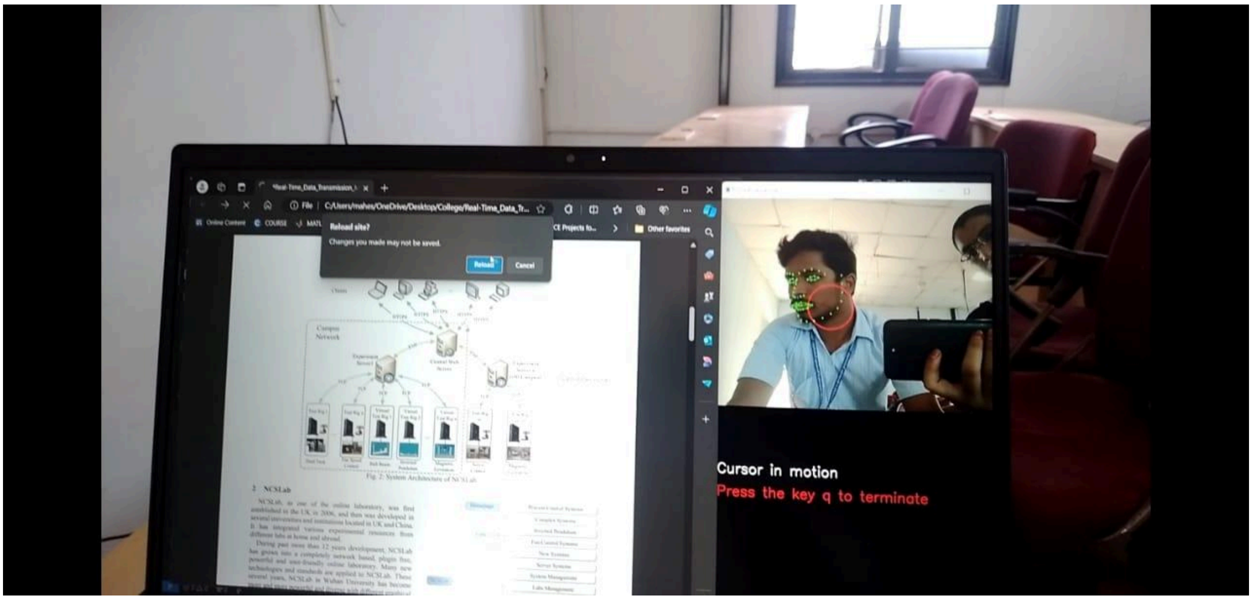


Figure 8 - Moving the cursor using the nose tip.

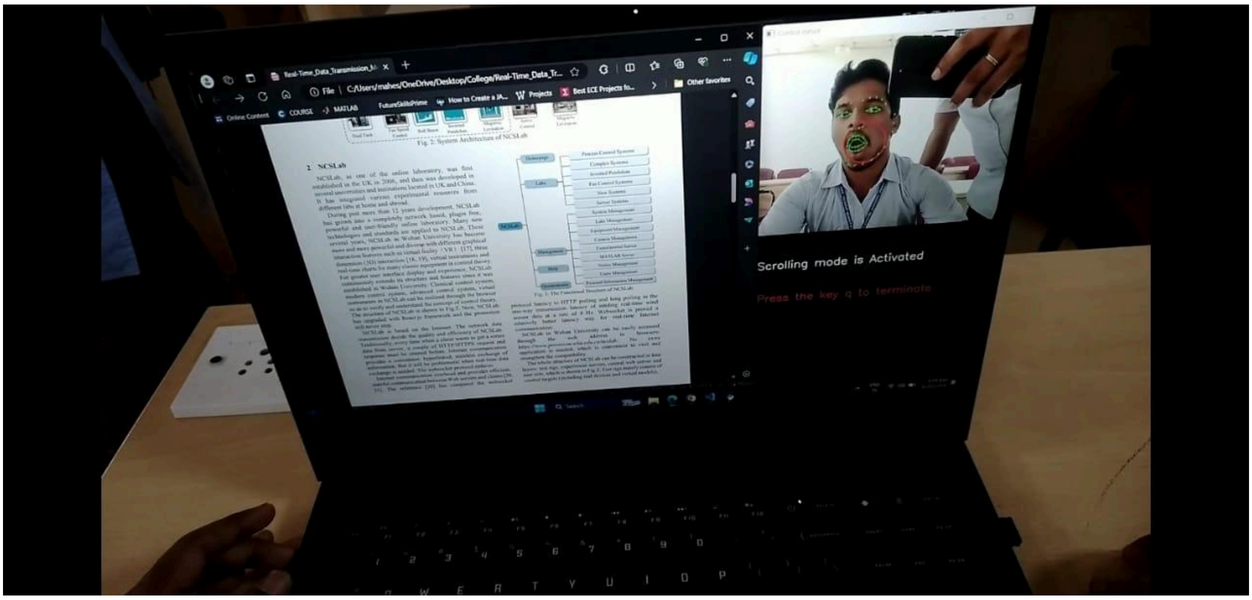


Figure 9 - Scrolling mode on/off by opening the mouth.

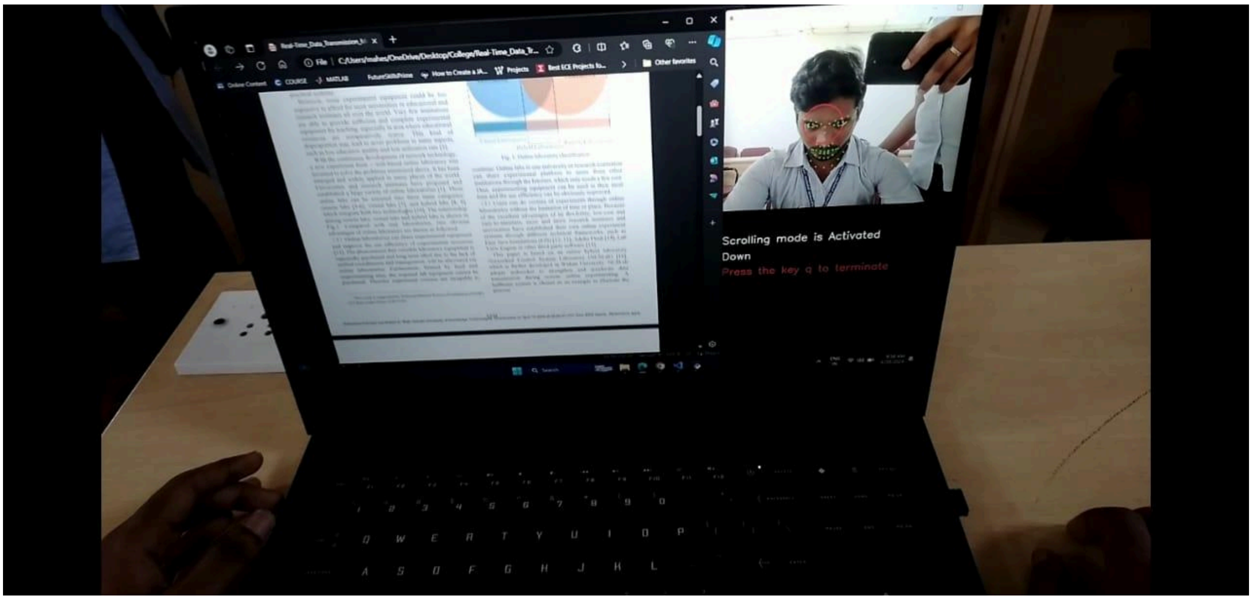


Figure 10 - Scrolling Up and Down using the nose tip.

CHAPTER-5

CONCLUSION

This research attempts to make computer interaction more accessible to physically challenged persons by employing face motions to control mouse activities and monitor the screen. The programme captures photographs using a camera and processes facial motions for identification. This unique idea has the potential to significantly enhance the experience of physically challenged folks who struggle to operate a regular mouse. It is a step towards making technology more inclusive and accessible to everybody.

CHAPTER-6

REFERENCES

1. Shashidhar R, Snehith K, Abhishek P K, Abhishek B Vishwagna, Pavitha N(2022) “Mouse Cursor Control Using Facial Movements - An HCI Application”
2. Salsabiil Hasanah, Aulia Teaku Nururrahmah, Darlis Herumurti(2021) “Comparative Analysis of Hands-free Mouse Controlling based on Face Tracking”
3. Palaniappan M, Sowmia K R, Aravindkumar S(2022) “Real Time Fatigue Detection Using Shape Predictor 68 Face Landmarks Algorithm”
4. Aleksandra Królak, Pawel Strumillo(2011) “Eye-blink detection system for human–computer interaction”

CHAPTER-7

SOURCE CODE

```
from imutils import face_utils
import dlib
import cv2
import pyautogui as pag
import numpy as n
import time

def Mar(p1,p2,p3,p4,p5,p6,p7,p8):
	mar = (dst(p1,p2) + dst(p3,p4) + dst(p5,p6))/(3.0*dst(p7,p8))
	return mar

def Ear(p1,p2,p3,p4,p5,p6):
	ear = (dst(p2,p6) + dst(p3,p5))/(2*dst(p1,p4))*1.0
	return ear
def dst(p1,p2):
	dist = n.sqrt((p1[0] - p2[0])**2 + (p1[1] - p2[1])**2)
	return dist

def angle(p1):
	slp = (p1[1] - 250)/(p1[0] - 250)*1.0
	agle = 1.0*n.arctan(slp)
	return agle


leclick = n.array([])
riclick = n.array([])
scroll= n.array([])
eye__open = n.array([])
leclickarea = n.array([])
riclickarea = n.array([])
pag.PAUSE =0
y= "shape_predictor_68_face_landmarks.dat"
det = dlib.get_frontal_face_detector()
pred= dlib.shape_predictor(y)
(l1start,l1end)   =   face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]
(r1start,r1end)   =   face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
(m1start,m1end) = face_utils.FACIAL_LANDMARKS_IDXS["mouth"]

cp= cv2.VideoCapture(0)
ft = cv2.FONT_HERSHEY_SIMPLEX
ctime = time.time()
while(time.time() - ctime <= 25):
	r,img = cp.read()
```

```
        bimg = n.zeros((480,640,3),dtype = n.uint8)
        img = cv2.flip(img,1)
        gry = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
        cl = cv2.createCLAHE(clipLimit = 2.1, tileGridSize = (8,8))
        gry = cl.apply(gry)
        rets = det(gry,0)
        for (i,ret) in enumerate(rets ):
                shp = pred(gry,ret)
                shp = face_utils.shape_to_np(shp)
                leye = Ear(shp[36],shp[37],shp[38],shp[39],shp[40],shp[41])
                reye = Ear(shp[42],shp[43],shp[44],shp[45],shp[46],shp[47])
                mar = Mar(shp[50],shp[58],shp[51],shp[57],shp[52],shp[56],shp[48],shp[54])
                EYE_DIFF = (leye - reye)*100
                ll__Eye = shp[l1start:l1end]
                rr__Eye = shp[r1start:r1end]
                mroi = shp[m1start:m1end]
                lEyehull = cv2.convexHull(ll__Eye)
                rEyehull = cv2.convexHull(rr__Eye)
                mthull = cv2.convexHull(mroi)
                cv2.drawContours(img,[mroi],-1,(0,255,60),1)
                cv2.drawContours(img,[lEyehull],-1,(0,255,60),1)
                cv2.drawContours(img,[rEyehull],-1,(0,255,60),1)
                Mouarea = cv2.contourArea(mthull)
                lefarea = cv2.contourArea(lEyehull)
                Rigarea = cv2.contourArea(rEyehull)
                etime = time.time() - ctime
                if etime < 5.0:
                        cv2.putText(bimg,'Open Eyes',(0,100), ft,
1,(245,245,245),2,cv2.LINE_AA)
                        eye__open = n.append(eye__open,[EYE_DIFF])
                elif etime > 5.0 and etime < 10.0:
                        cv2.putText(bimg,'left eye close',(0,100), ft,
1,(245,245,245),2,cv2.LINE_AA)
                        leclick = n.append(leclick,[EYE_DIFF])
                        leclickarea = n.append(leclickarea,[lefarea])
                elif etime > 11.0 and etime < 18.0:
                        cv2.putText(bimg,'close right eye and open left eye',(0,100), ft,
1,(245,245,245),2,cv2.LINE_AA)
                        riclick = n.append(riclick,[EYE_DIFF])
                        riclickarea = n.append(riclickarea,[Rigarea])
                elif etime > 19.0 and etime < 24.0:
                        cv2.putText(bimg,'Mouth
open',(0,100),ft,1,(245,245,245),2,cv2.LINE_AA)
                        if etime > 21.0 and etime < 24.0:
                                scroll = n.append(scroll,[mar])
                else:
                        pass
                for (x,y) in shp:
                        cv2.circle(img,(x,y),2,(0,255,60),-1)
                out = n.vstack((img,bimg))
                cv2.imshow('CALIBRATION:',out)
```

```
        if cv2.waitKey(5) & 0xff == 113:
                break
cp.release()
cv2.destroyAllWindows()

cp = cv2.VideoCapture(0)
MAR = n.array([])
sc_sts = 0
eye__open = n.sort(eye__open)
leclick = n.sort(leclick)
riclick = n.sort(riclick)
scroll = n.sort(scroll)
leclickarea = n.sort(leclickarea)
riclickarea = n.sort(riclickarea)
openeyes = n.median(eye__open)
LEFT_CLICK = n.median(leclick) - 1
RIGHT_CLICK = n.median(riclick) + 1
scrllng = n.median(scroll)
LEFTCLICK_AREA = n.median(leclickarea)
RIGHTCLICK_AREA = n.median(riclickarea)

while(True):
        try:
                bimg = n.zeros((480,640,3),dtype = n.uint8)
                __, img = cp.read()
                img=cv2.flip(img,1)
                gry = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
                cl = cv2.createCLAHE(clipLimit = 2.1, tileGridSize = (8,8))
                gry = cl.apply(gry)
                cv2.circle(img,(250,250),50,(0,0,255),2)

                rets = det(gry, 0)

                for (i, ret) in enumerate(rets ):

                        shp = pred(gry, ret)
                        shp = face_utils.shape_to_np(shp)
                        [h,k] = shp[33]
                        leye = Ear(shp[36],shp[37],shp[38],shp[39],shp[40],shp[41])
                        reye = Ear(shp[42],shp[43],shp[44],shp[45],shp[46],shp[47])
                        EYE_DIFF = (leye - reye )*100
                        mar =
Mar(shp[50],shp[58],shp[51],shp[57],shp[52],shp[56],shp[48],shp[54])
                        cv2.line(img,(250,250),(h,k),(0,0,0),1)
                        ll__Eye = shp[l1start:l1end]
                        rr__Eye = shp[r1start:r1end]
                        mroi = shp[m1start:m1end]
                        lEyehull = cv2.convexHull(ll__Eye)
                        rEyehull = cv2.convexHull(rr__Eye)
                        mthull = cv2.convexHull(mroi)
                        cv2.drawContours(img,[mroi],-1,(0,255,60),1)
```

```
                    cv2.drawContours(img,[lEyehull],-1,(0,255,60),1)
                    cv2.drawContours(img,[rEyehull],-1,(0,255,60),1)
                    lefarea = cv2.contourArea(lEyehull)
                    Rigarea = cv2.contourArea(rEyehull)
                    Mouarea = cv2.contourArea(mthull)
                    if EYE_DIFF < LEFT_CLICK and lefarea < LEFTCLICK_AREA:
                        pag.click(button = 'Left')

    cv2.putText(bimg,"LEFTCLICK",(0,100),ft,1,(245,245,245),2,cv2.LINE_AA)
                        leclick = n.array([])
                    elif EYE_DIFF > RIGHT_CLICK and Rigarea < RIGHTCLICK_AREA:
                        pag.click(button = 'Right')

    cv2.putText(bimg,"RIGHTCLICK",(0,100),ft,1,(245,245,245),2,cv2.LINE_AA)
                        leclick = n.array([])

                for (x, y) in shp:
                    cv2.circle(img, (x, y), 2, (0, 245, 1), -1)
                MAR = n.append(MAR,[mar])
                if len(MAR) == 30:
                    mar_avg = n.mean(MAR)
                    MAR = n.array([])
                    if int(mar_avg*100) > int(scrllng*100):
                        if sc_sts == 0:
                            sc_sts = 1
                        else:
                            sc_sts = 0

                if sc_sts == 0:
                    if((h-250)**2 + (k-250)**2 - 50**2 > 0):
                        a = angle(shp[33])
                        if h > 250:
                            time.sleep(0.03)
                            x = 10*n.cos(1.0*a)
                            y = 10*n.sin(1.0*a)

    pag.moveTo(pag.position()[0]+(x),pag.position()[1]+(y),duration = 0.01)
                            cv2.putText(bimg,"Cursor in
motion",(0,150),ft,1,(245,245,245),2,cv2.LINE_AA)
                        else:
                            time.sleep(0.03)
                            x = 10*n.cos(1.0*a)
                            y = 10*n.sin(1.0*a)
                            pag.moveTo(pag.position()[0]-(x),pag.position()[1]-
(y),duration = 0.01)
                            cv2.putText(bimg,"Cursor in
motion",(0,150),ft,1,(245,245,245),2,cv2.LINE_AA)
                else:
                    cv2.putText(bimg,'Scrolling mode is
Activated',(0,100),ft,1,(245,245,245),2,cv2.LINE_AA)
                    if k > 300:
```

```
                                        cv2.putText(bimg,"Down
",(0,150),ft,1,(245,245,245),2,cv2.LINE_AA)
                                        pag.scroll(-12)
                              elif k < 200:
                                        cv2.putText(bimg,"Up
",(0,150),ft,1,(245,245,245),2,cv2.LINE_AA)
                                        pag.scroll(12)
                              else:
                                        pass
                    cv2.circle(img,(h,k),2,(245,1,0),-1)
                    cv2.putText(bimg,"Press the key q to
terminate",(0,200),ft,1,(0,10,245),2,cv2.LINE_AA)
                    out = n.vstack((img,bimg))
                    cv2.imshow('Control cursor',out)
                    r = cv2.waitKey(5) & 0xFF
                    if r == 113:
                              break
          except:
                    bimg = n.zeros((480,640,3),dtype = n.uint8)
                    cv2.putText(bimg,"No Landmark is
detected",(0,100),ft,1,(0,10,245),2,cv2.LINE_AA)
                    __,img = cp.read()
                    img= cv2.flip(img,1)
                    res = n.vstack((img,bimg))
                    cv2.imshow('Cursor Control',res)
                    r = cv2.waitKey(5) & 0xff
                    if r == 113:
                              break

cv2.destroyAllWindows()
cp.release()
```