

EYE TRACKING-BASED CONTROL SYSTEM FOR NATURAL HUMAN-COMPUTER INTERACTION

A Project Report

Submitted to the Faculty of Engineering of

**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY KAKINADA,
KAKINADA.**

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

BACHELOR OF TECHNOLOGY
In
INFORMATION TECHNOLOGY

By

D. LAKSHMI MANASA
(19481A1225)

G. GOPI KRISHNA REDDY
(19481A1231)

M. NIKITHA
(20485A1204)

B. SAI KIRAN
(19481A1209)

Under the Enviabale and Esteemed Guidance of

Mr. B. SOBHAN BABU, M.Tech., (ph.D)

Assistant Professor, Department of IT



DEPARTMENT OF INFORMATION TECHNOLOGY

SESHADRI RAO GUDLAVALLERU ENGINEERING COLLEGE

(An Autonomous Institute with Permanent Affiliation to JNTUK, Kakinada)

SESHADRI RAO KNOWLEDGE VILLAGE

GUDLAVALLERU-521356

ANDHRA PRADESH

2022-23

DEPARTMENT OF INFORMATION TECHNOLOGY

SESHADRI RAO GUDLAVALLERU ENGINEERING COLLEGE

(An Autonomous Institute with Permanent Affiliation to JNTUK, Kakinada)

SESHADRI RAO KNOWLEDGE VILLAGE

GUDLAVALLERU-521356



CERTIFICATE

This is to certify that the project report entitled “**EYETRACKING BASED CONTROL SYSTEM FOR NATURAL HUMAN-COMPUTER INTERACTION**” is a bonafide record of work carried out by **D. Lakshmi Manasa (19481A1225)**, **G. Gopi Krishna Reddy (19481A1231)**, **M. Nikitha (20485A1204)**, **B. Sai Kiran(19481A1209)** under the guidance and supervision of **Mr. B. SOBHAN BABU** in the partial fulfillment of there requirements the award of the degree of Bachelor of Technology in Information Technology and Engineering of **Jawaharlal Nehru Technological University Kakinada, Kakinada** during the academic year 2022-23.

Project Guide

Mr. B. SOBHAN BABU

Head of the Department

Dr. CH. KAVITHA

External Examiner

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of people who made it possible and whose constant guidance and encouragements crown all the efforts with success.

We would like to express our deep sense of gratitude and sincere thanks to **Mr. B. Sobhan Babu, Assistant professor**, Department of Information technology for his constant guidance, supervision and motivation in completing the project work.

We feel elated to express our floral gratitude and sincere thanks to **Dr. Ch. Kavitha**, Head of the Department, Information Technology for her encouragement all the way during analysis of the project. Her annotations, insinuations and criticisms are the key behind the successful completion of the project work.

We would like to take this opportunity to thank our beloved principal **Dr. G. V. S. N. R. V. Prasad** for providing a great support for us in completing our project and giving us the opportunity for doing project.

Our Special thanks to the faculty of our department and programmers of our computer lab. Finally, we thank our family members, non-teaching staff and our friends, who had directly or indirectly helped and supported us in completing our project in time.

Team members

D. Lakshmi Manasa (19481A1225)
G. Gopi Krishna(19481A1231)
M. Nikitha (20485A1204)
B. Sai Kiran (19485A1209)

ABSTRACT

There is consistently a potential open door for improvement in the field of PCs, particularly with the present complex advancements without hands registering has been well known as of late because of the requirements of quadriplegics. This examination presents a captivating human-PC collaboration (HCI)frame work. This would be incredibly advantage ous to tragically handicapped people and people Who experience issues utilizing their hands. Flickering, looking, and squinting are movements of every Kind focused on the mouse pointer, and the framework set up is an eye-based interface that fills in as a PC mouse to interpret eye developments like flickering, gazing, and squinting. The product prerequisites for the framework being referred to are Python (3.6), OpenCV, and a fundamental camera. Numpy and a couple of more libraries are expected for facial acknowledgment. Consolidating the HOG (Histogram of Oriented Gradients) highlight with a straight change can be utilized to foster a facial acknowledgment framework. A classifier with a sliding window procedure It's completely without hands and requires no extra equipment or programming.

INDEX

TITLE	PAGE NO
CHAPTER 1: INTRODUCTION	1-2
1.1 INTRODUCTION	1
1.2 PROBLEM STATEMENT	1
1.3 EXISTING SYSTEM	2
1.4 DISADVANTAGES	2
1.5 PROPOSED SYSTEM	2
1.6 ADVANTAGES	2
CHAPTER 2: REQUIREMENT ANALYSIS	3
2.1 FUNCTIONAL REQUIREMENTS	3
2.2 NON- FUNCTIONAL REQUIREMENTS	3
2.3 SOFTWARE REQUIREMENT SPECIFICATIONS	3
2.4 HARDWARE SPECIFICATIONS	3
CHAPTER 3: DESIGN	4-8
3.1 SYSTEM ARCHITECTURE	5
3.2 UML DIAGRAMS	5
3.3 USE CASE DIAGRAM	6
3.4 CLASS DIAGRAM	7
3.5 SEQUENCE DIAGRAM	7
3.6 ACTIVITY DIAGRAM	8

CHAPTER 4: IMPLEMENTATION	9-17
4.1 TECHNOLOGY DESCRIPTION	9
4.2 INSTALLATION STEPS	10
4.3 PROCEDURE FOR EXECUTION	12
CHAPTER 5: TESTING	18-25
5.1 LEVEL SOFTWARE TESTING	18
5.2 UNIT TESTING	18
5.3 INTEGRATION TESTING	19
5.4 SYSTEM TESTING	19
5.5 BLACK BOX TESTING	19
5.6 WHITE BOX TESTING	20
CHAPTER 6: SCREENSHOTS	26-31
CHAPTER 7: CONCLUSION	32
REFERENCES	33
MAPPING OF COs AND POs	34

LIST OF FIGURES

Figure No	Name of the figure	Page No
Fig 3.1.1	Design Methodology	1
Fig 3.2.1	Use case Diagram	5
Fig 3.2.2	Class Diagram	6
Fig 3.2.3	Sequence Diagram	7
Fig 3.2.4	Activity Diagram	8
Fig 4.1.1	Installation of visual Studio	9-10
Fig 4.2.1	File Opening	13
Fig 4.2.2	Run the Code	14
Fig 4.2.3	Code Execution	14
Fig 6.1.1	Reading Input	27
Fig 6.1.2	Scroll mode on	28
Fig 6.1.3	Scrolling Right	28
Fig 6.1.4	Scrolling Left	29
Fig 6.1.5	Scrolling Up	29
Fig 6.1.6	Scrolling Down	30
Fig 6.1.7	Right movement of cursor	30
Fig 6.1.8	Left movement of cursor	31

CHAPTER-1

INTRODUCTION

CHAPTER -1

INTRODUCTION

1.1 INTRODUCTION

Using a PC mouse or shifting the finger to move the cursor around the screen is a standard method inside the gift innovation. The framework recognises and correlates any mouse or finger improvement to cursor improvement. Positive persons who are unable to use their arms, referred to as "tragically handicapped people," will no longer be able to operate the mouse as a result of modern invention. As a result, if the improvement in their eyeball and the direction in which their attention is observing can be separated, the improvement in their eyeball is almost certainly purposeful to the cursor, permitting the terribly disabled individual to transport the pointer at him. To a horribly handicapped person, an 'eye following mouse' could be quite valuable. The eye is no longer commonly available to view the mouse because it is no longer a gift. We want to create an eye-fixed following mouse that has a lot of the functionality of a standard mouse but also allows the user to move the pointer with his eye. We aim to anticipate the customer's 'gaze' and circulate the mouse accordingly. For quite some time, the mouse's pointing and clicking have remained the same old. Certain people may find them unpleasant for a variety of reasons, and those who cannot use their palms may have to use a mouse without palms. The foundation for a mouse without palms is for the most important eye and face gestures.

1.2 Problem Statement:

Hardware cost is expensive. It is difficult when there is no space to use a physical mouse. The person who has problems in their hands and is not able to control the physical mouse.

The Covid-19 situation, may result in a possible situation of spread of the virus by touching the devices.

1.3 Existing System:

Some researchers have been attempting to establish techniques that help the elderly communicate With devices. Many high-end techniques focused on eye movement monitoring to monitor computers were extremely costly and not available to those who wanted them. In some available software, it difficult to predict the centroid of the eye so we go for OpenCV.

1.4 Disadvantages:

1. Less accuracy
2. NUnstable
3. Required prior knowledge for using

1.5 Proposed System:

In the proposed system, we have included face detection, face tracking, eye detection and interpretation of a sequence of eye blinks in real time for controlling a nonintrusive human-computer interface. A conventional method of interaction with the computer with the mouse is replaced with human eye movements. This technique will help the paralyzed person, physically challenged people especially people without hands to compute efficiently and with ease of use. Firstly, the camera captures the image and focuses on the eye in the image using Open CV code for pupil detection. This results in the center position of the human eye (pupil). Then the center position of the pupil is taken as a reference and based on that the human or the user will control the cursor by moving left and right. This paper's organization is describedas follows. Section II describes existing solutions to find the cursor movement using some 3Dmodels. In Section III we present how the cursor is working based only on Eyeball movement using OpenCV methodology. In Section IV how the cursor is moving using eyeball with examples with the better solutions. And the Conclusion part is presented in section V.

1.6 Advantages:

- It is difficult when there is no space to use a physical mouse.
- The person who have problems in their hands and are not able to control physical mouse.
- In Covid-19 situation, it may result in a possible situation of spread of the virus by touching the devices
- Physically handicapped people can operate computers.

CHAPTER-2

REQUIREMENT ANALYSIS

CHAPTER - 2

REQUIREMENT ANALYSIS

2.1 FUNCTIONAL REQUIREMENTS

- Numpy-1.13.3
- OpenCv-3.2.0
- PyAutoGUI-0.9.36
- Dlib-19.4.0
- Imutils-0.4.6

2.2 NON-FUNCTIONAL REQUIREMENTS

- High Performance
- High Maintainability
- Highly Reliability

2.3 SOFTWARE REQUIREMENT SPECIFICATIONS

- **Tool Used** : Anaconda Tool, Visual Studio
- **IDE** : Python IDE, Command Prompt
- **Programming Language** : Python3
- **Operating System** : Windows

2.4 HARDWARE SPECIFICATIONS

- **Processor** : i3
- **RAM** : 4GB
- **Hard Disk** : 1TB
- **Webcam/Integrated cam.**

CHAPTER-3

DESIGN

CHAPTER - 3

DESIGN

3.1 System Architecture:

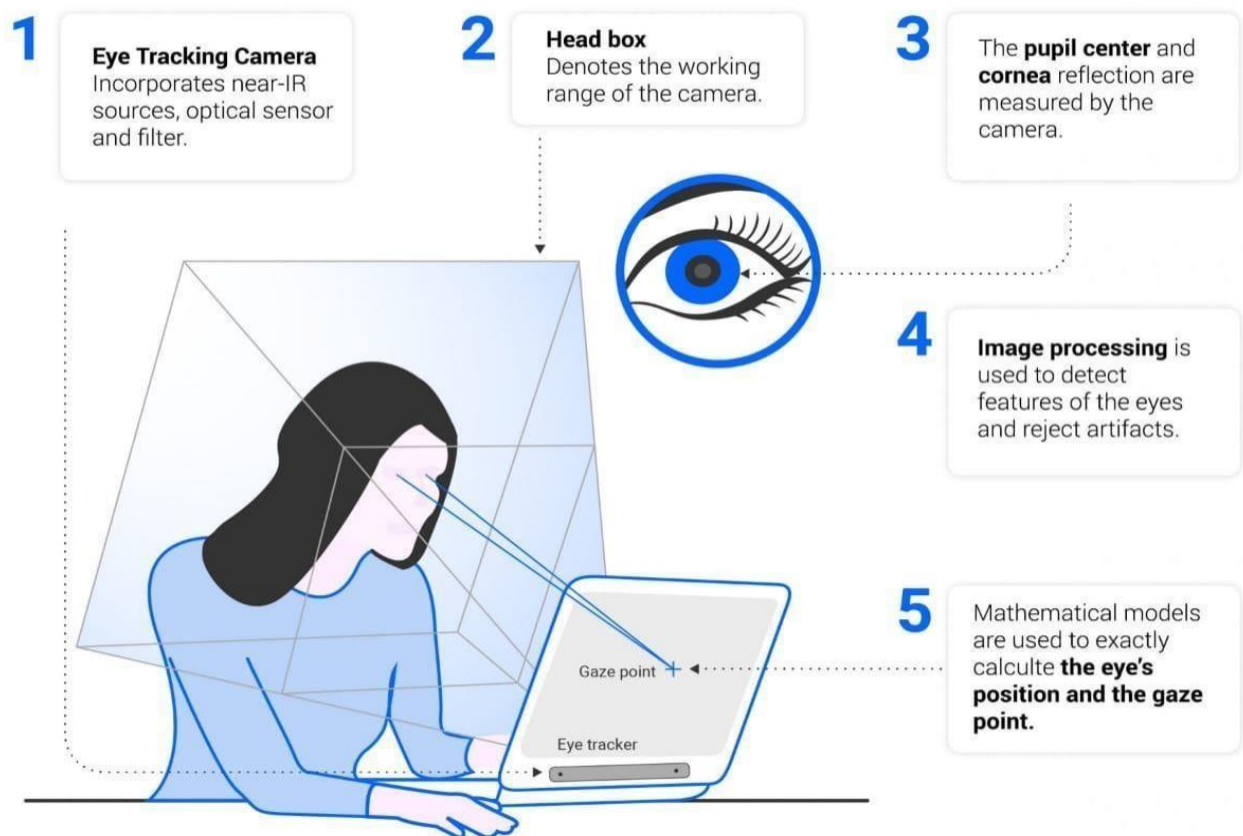


Fig 3.1.1: Design Methodology

The Software Testing Life Cycle is the systematic and planned process of software testing (STLC). Distinct firms have different STLC stages, however the waterfall development model's common SoftwareTest Life Cycle (STLC) is as follows:

- ★ Need analysis
- ★ Test planning
- ★ Test Analysis

3.2 UML DIAGRAMS

3.2.1 Use Case Diagram

The purpose of use case diagram is to capture the dynamic aspect of a system. However, this definition is too generic to describe the purpose, as other four diagrams (activity, sequence, collaboration, and state chart) also have the same purpose. We will look into some specific purpose, which will distinguish it from the other four diagrams.

Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. Hence, when a system is analyzed together its functionalities, and use cases are prepared and actors are identified.

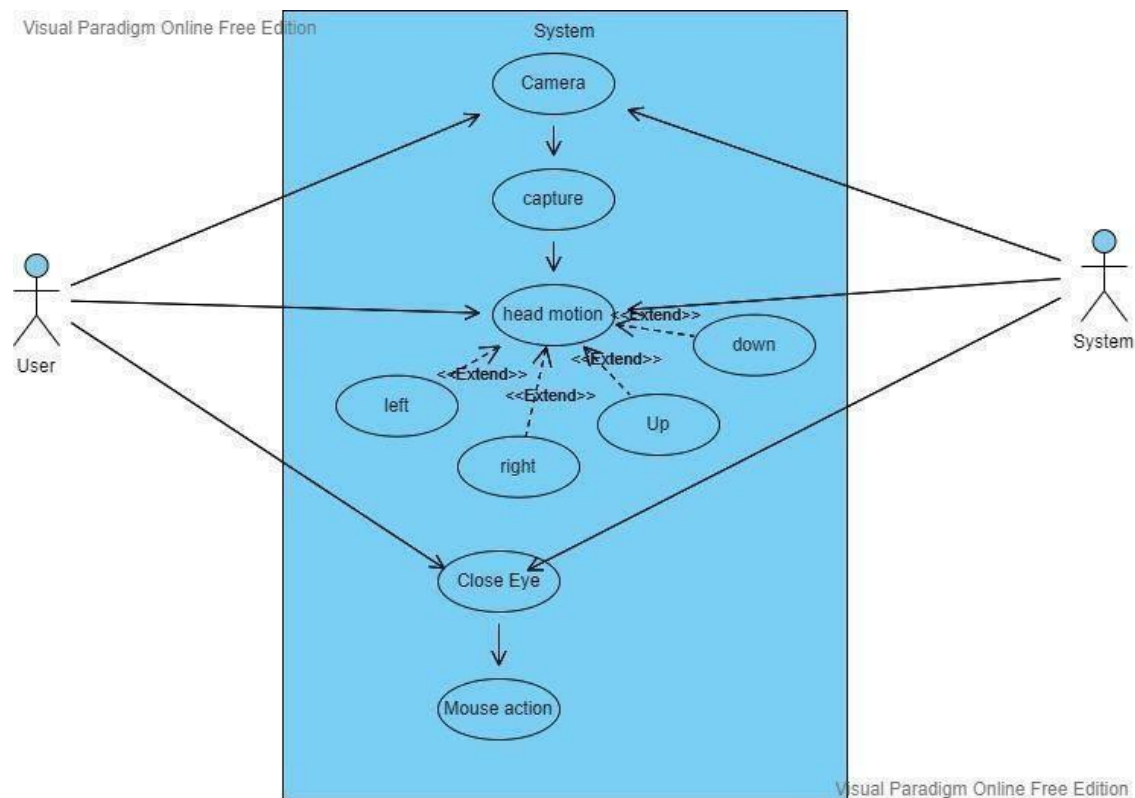


Fig: 3.2.1 Use case diagram

3.2.2 Class Diagram

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object-oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages. Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.

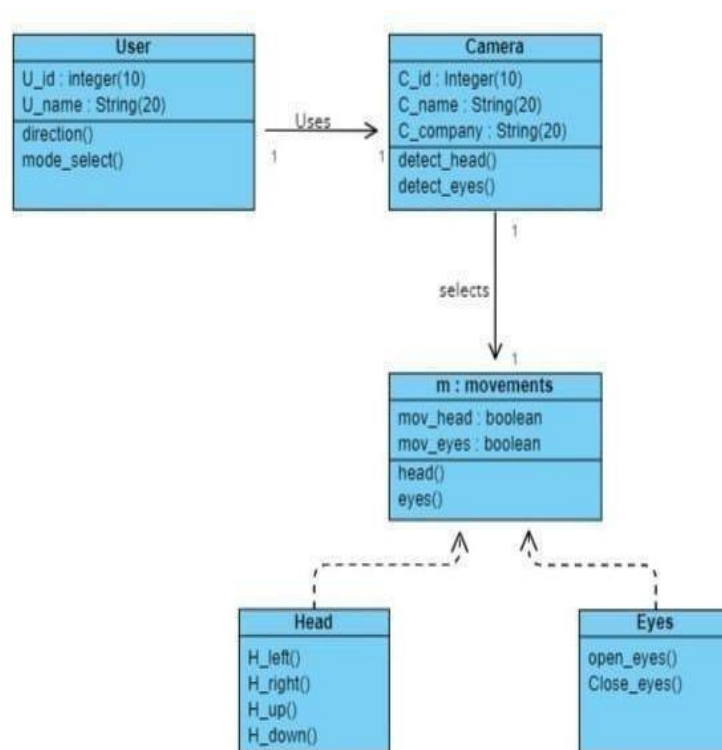


fig. 3.2.2 Class diagram

3.2.3 Sequence Diagram

The sequence diagram represents the flow of messages in the system and is also termed as an event diagram. It he linen visioning several dynamic scenarios. It portrays the communication between any two lifelines as a time-ordered sequence of events, such that these lifelines took part in the run time. In UML, the lifeline is represented by a vertical bar, whereas the message flow is represented by a vertical dotted line that extends across the bottom of the page. It incorporates iterations as well as branching.

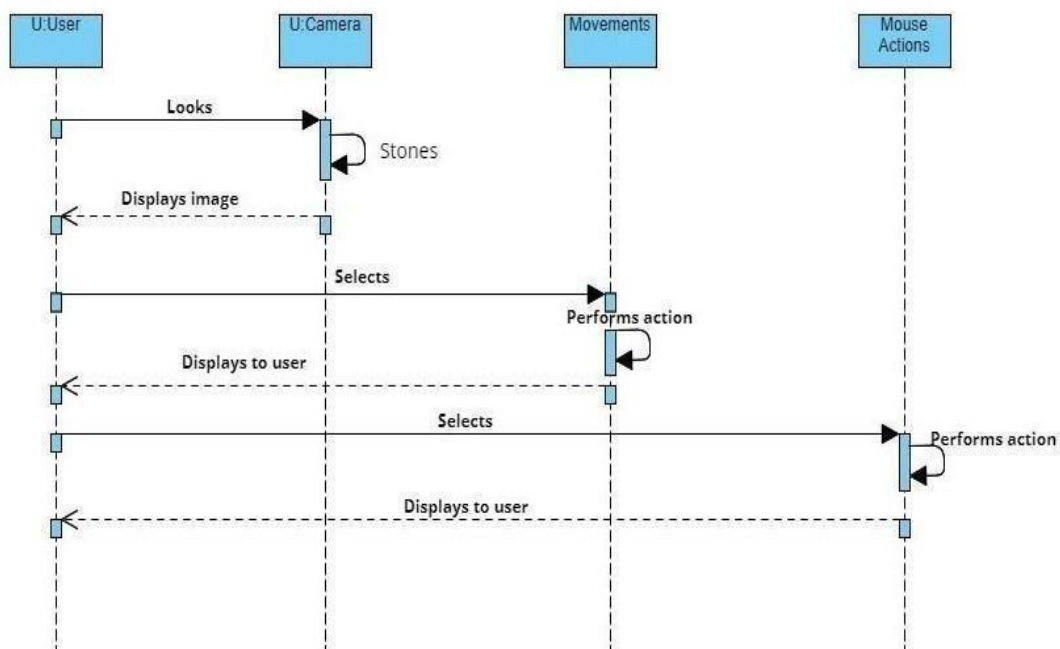


Fig 3.2.3: Sequence Diagram

3.2.4 Activity Diagram

An activity diagram visually presents a series of actions or flow of control in a system similar to a flowchart or a data flow diagram. Activity diagrams are often used in business process modeling. They can also describe the steps in a use case diagram. Activities modeled can be sequential and concurrent. In both cases, an activity diagram will have a beginning (an initial state) and an end (a final state). An activity diagram shows business and software processes as a progression of actions. These actions can be carried out by people, software components, or computers. Activity diagrams are used to describe business processes and use cases as well as to document the implementation of system processes.

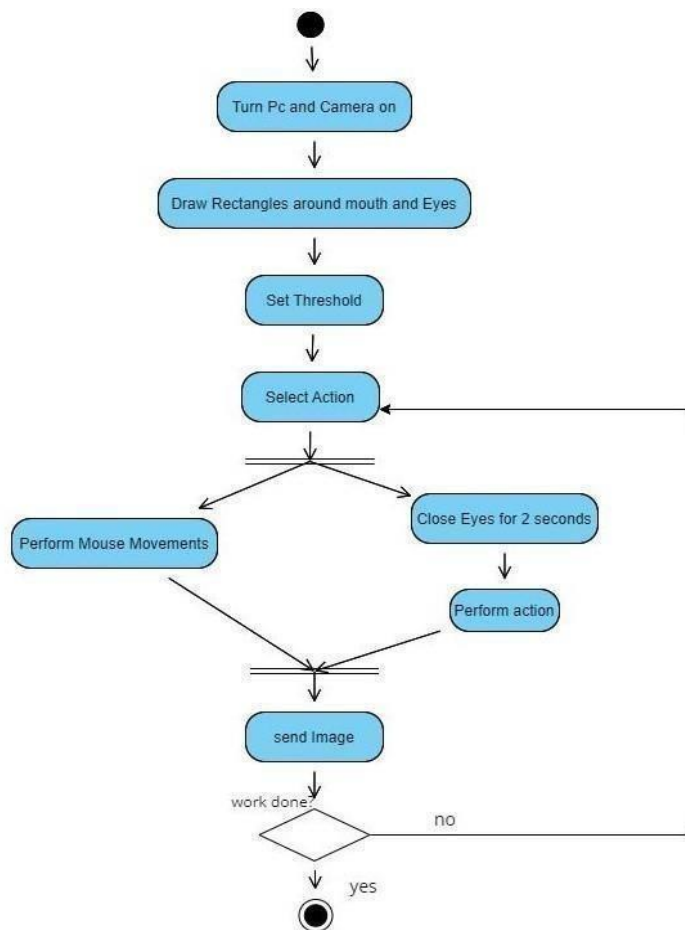


Fig 3.2.4: Activity Diagram

CHAPTER-4

IMPLEMENTATION

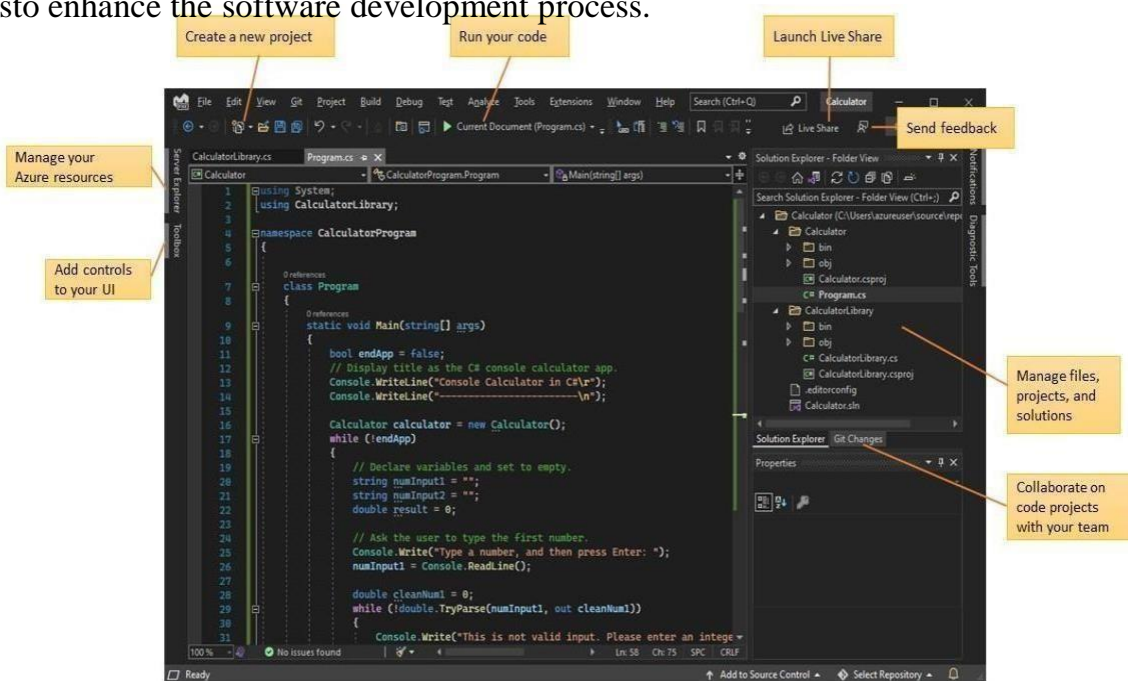
CHAPTER - 4

IMPLEMENTATION

4.1 TECHNOLOGY DESCRIPTION

Visual Studio:

An integrated development environment (IDE) is a feature-rich program that supports many aspects of software development. The Visual Studio IDE is a creative launching pad that you can use to edit, debug, and build code, and then publish an app. Over and above the standard editor and debugger that most IDEs provide, Visual Studio includes compilers, code completion tools, graphical designers, and many more features to enhance the software development process.



- The preceding image shows Visual Studio with an open project that shows there functionality
- In Solution Explorer, at upper right, you can view, navigate, and manage your code files. Solution Explorer can help organize your code by grouping the files into solutions and projects.
- The central editor window, where you'll probably spend most of your time, displays file contents. In the editor window, you can edit code or design a user interface such as a window with buttons and text boxes.

- In Git Changes at lower right, you can track work items and share code with others by using version control technologies like Git and GitHub.

4.2: Installation Steps

Install Visual Studio:

In this you create a simple project to try out some of the things you can do with Visual Studio. You use IntelliSense as a coding aid, debug an app to see a variable value during app execution, and change the color theme.

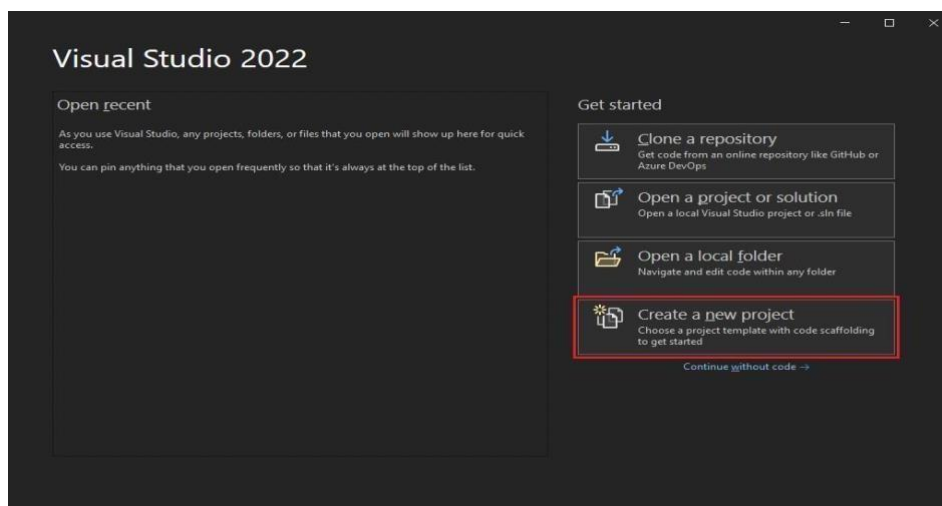
To get started, download Visual Studio and install it on your system. In the modular installer, you choose and install workloads, which are groups of features you need for the programming languages or platforms you want. To use the following steps to create a program, be sure to select the .NET desktop development workload during installation.

When you open Visual Studio for the first time, you can sign in by using your Microsoft account or your work or school account.

Create a program

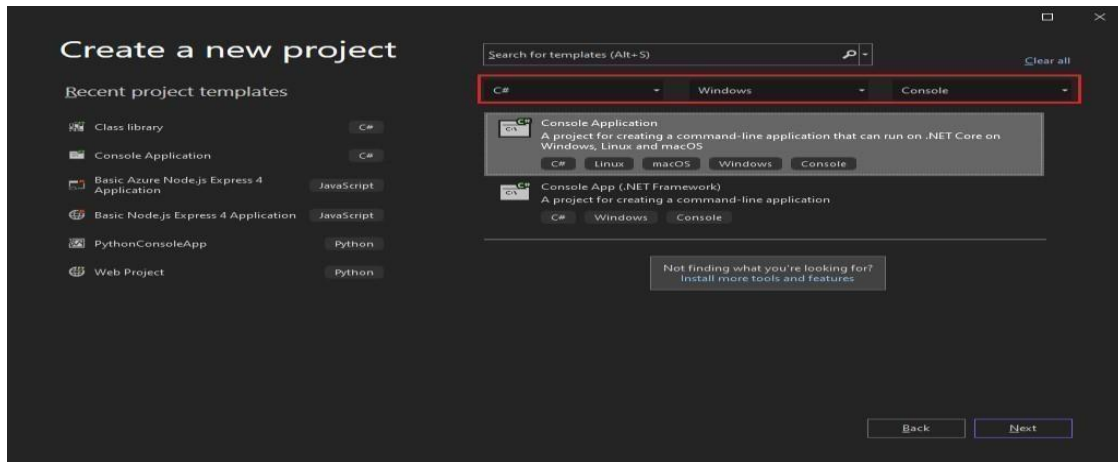
Dive in and create a simple program.

1. Start Visual Studio. The start window appears with options for cloning a repo, opening an existing project, or creating a new project.
2. Choose Create a new project.

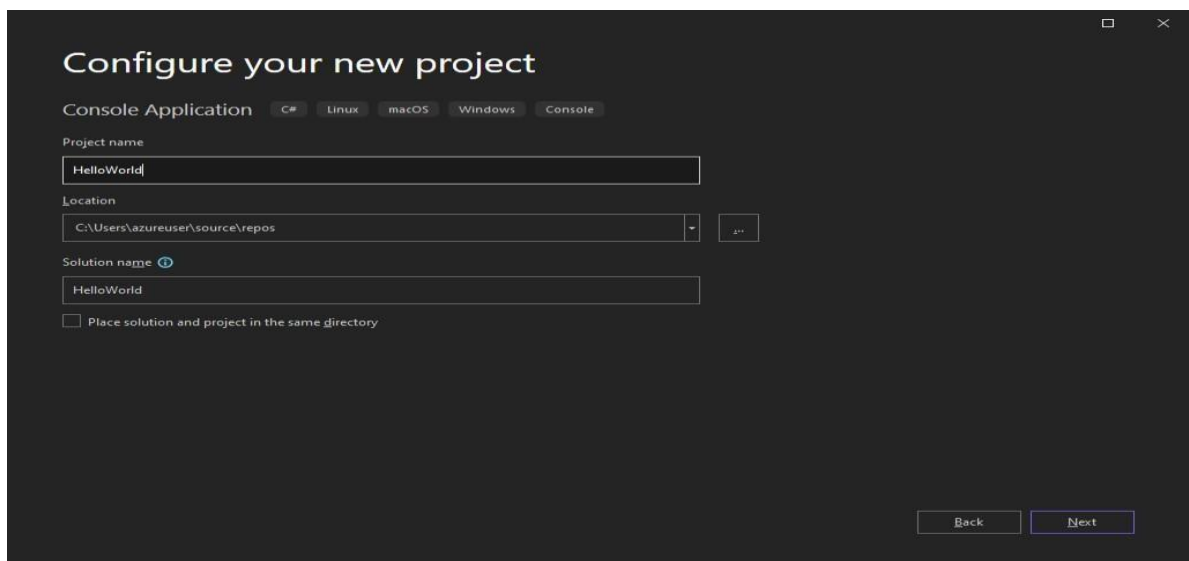


The Create a new project window opens and shows several project templates. A template contains the basic files and settings required for a given project type.

- To find a template, you can type or enter keywords in the search box. The list of available templates filters based on the keywords you enter. You can further filter the template results by choosing C# from the All languages drop-down list, Windows from the All platforms list, and Console from the All project type list. Select the Console Application template, and then select Next.



- In the Configure your new project window, enter HelloWorld in the Project name box. Optionally, change the project directory location from the default location of C:\Users\<name>\source\repos, and then select Next.



- In the Additional information window, verify that .NET 6.0 appears in the Target Framework drop-down menu, and then select Create.

4.3: Procedure for Execution

The following are the acts that make up the system proposed in this study:

- 4.3.1 Squinting your eyes
- 4.3.2 Winking
- 4.3.3 Head movement (pitch and yaw)
- 4.3.4 Opening your mouth

4.3 Requirements:

The requirements of this project have been listed, reviewed and discussed in detail.

We simply specify the software needed because it is the most important aspect of our project.

This is the following method:

Because the project relies on the webcam to recognise facial functionality and place it in the cursor, the webcam must first be available, which means it must be opened. When the camera is open, the program must restore each image of the video. A framework after 1/30 seconds will be handled because the video frame rate usually has about 30 frames per second. A set of procedures must be completed before the frame properties are defined and mapped to the pointer. This technique is repeated for each frame as part of a loop. Face areas need to be recognized after frame removal. As a result, the images go through a series of image processing processes so that the computer can recognize features such as eyes, mouth, and nose.

The processing techniques are:

1. Resize:

On the y-axis, the first image is rotated 90 degrees. After that, the image must be resized. You can set the image resolution into any value you choose to use the resolution technique, Depending on your needs. The new resolution of this project is 640 x 480.

2. Bgr to gray:

To generate more accurate results, the data we utilise to recognise the different areas of the face must be grey level. Therefore, RGB switches to the gray level of the image, meaning to say the frame of the camera's video, is required. The image can then be used to find and the face once it has been converted to gray scale.

3. Facial feature prediction and detection:

Face detection and prediction: The project uses a predefined model for face detection and features, which includes built-in variables that Python can read to confirm that the face is there in the image. Face detection is handled by a method named "detector()" in models.

The "prediction" tool can now be used to find facial features after a face has been recognized. The application can be used to place 68 points on any two-dimensional image. These points correspond to a number of locations on the face, including those near the eyes, lips, and other

significant of 2D coordinates. Each of the 68 points is nothing more than a collection of x and y coordinate values that, when combined together, make a roughly identifiable face. They are then sorted and used to bind anycoordinates and build contours to depict the required portions of the face in the next stage. Four sets of panels indicate the required areas: left eye, right eye, nose, and mouth. Contours or "contours" are formedaround the points using three of them and connecting them using the "draw contour" tool, making a shapearound the eyes and from the mouth.

4. Mouth and eye aspect ratio:

Perspective ratio of mouth and eyes: after framing, the program needs a reference for shapes that, when contrasted, illuminate the product by any movement directed by these areas, such as blinking.

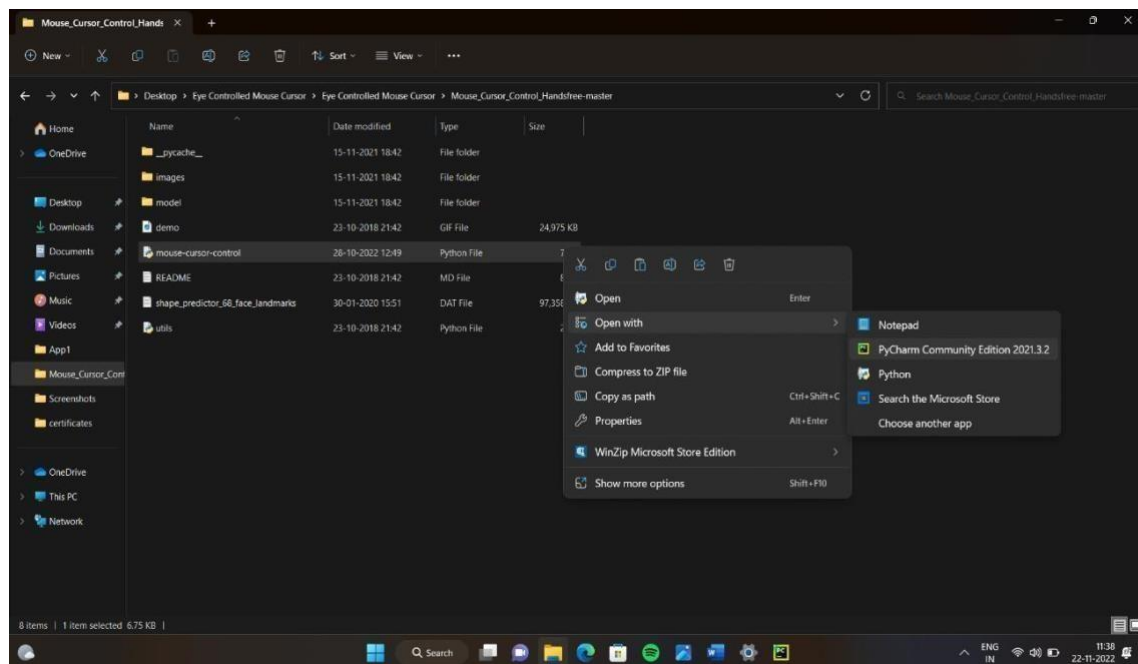
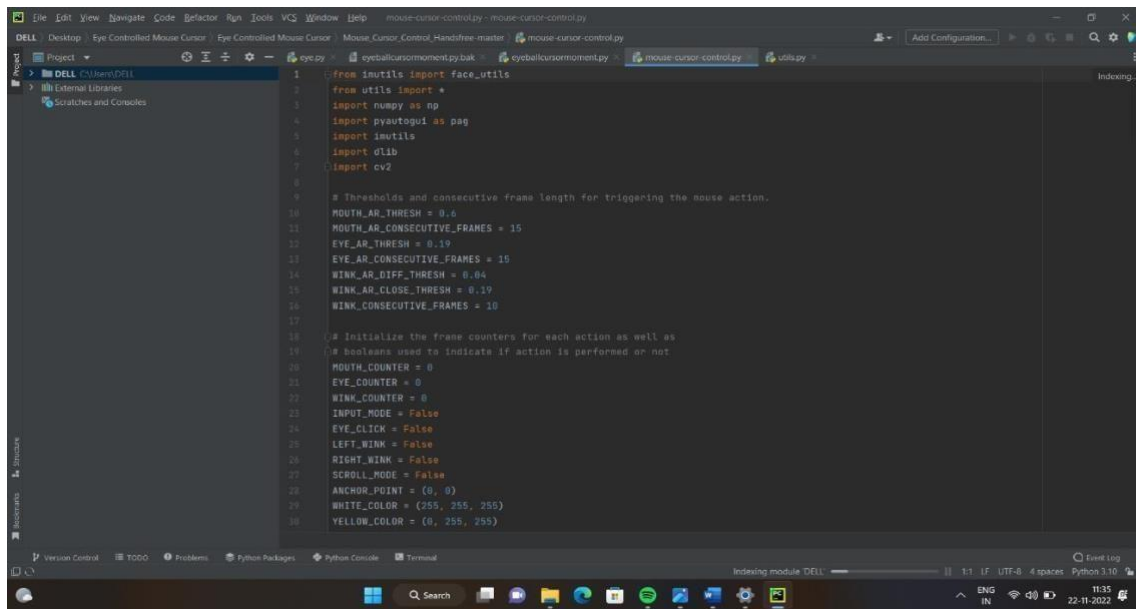


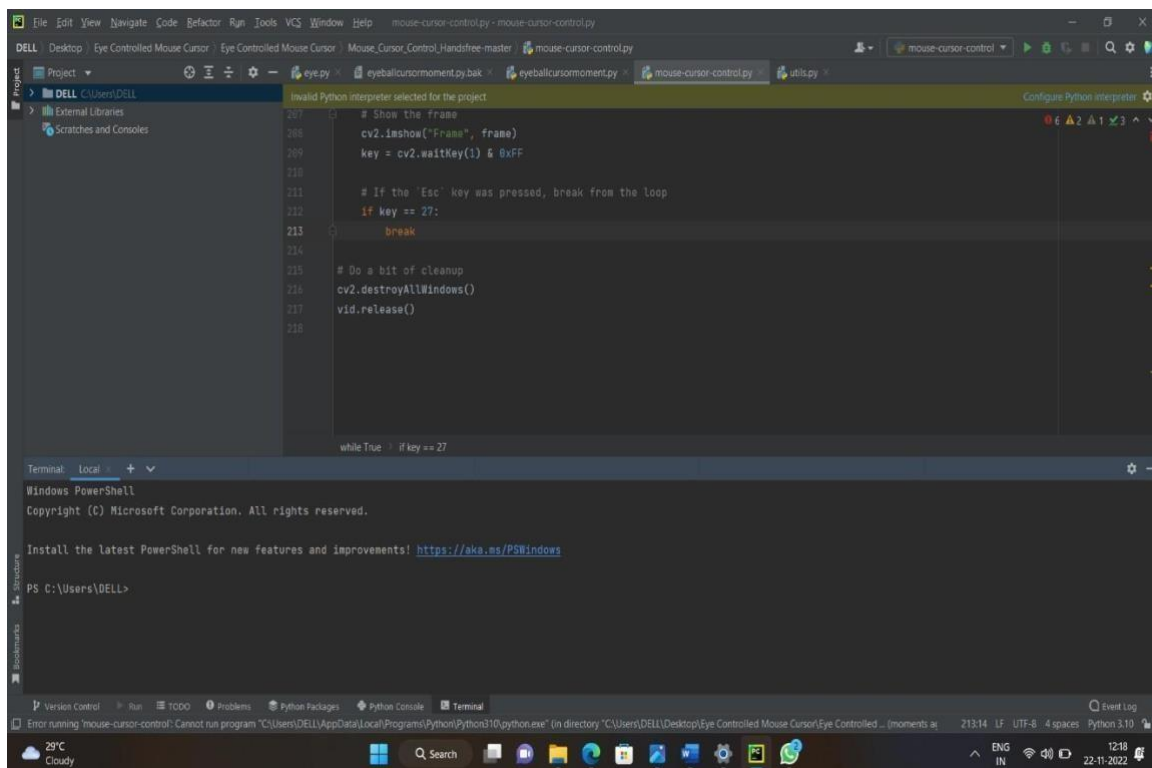
Fig: File Opening



```

1 from imutils import face_utils
2 from utils import *
3 import numpy as np
4 import pyautogui as pag
5 import imutils
6 import dlib
7 import cv2
8
9 # Thresholds and consecutive frame length for triggering the mouse action.
10 MOUTH_AR_THRESH = 0.6
11 MOUTH_AR_CONSECUTIVE_FRAMES = 15
12 EYE_AR_THRESH = 0.19
13 EYE_AR_CONSECUTIVE_FRAMES = 15
14 WINK_AR_DIFF_THRESH = 0.04
15 WINK_AR_CLOSE_THRESH = 0.19
16 WINK_CONSECUTIVE_FRAMES = 10
17
18 # Initialize the frame counters for each action as well as
19 # booleans used to indicate if action is performed or not
20 MOUTH_COUNTER = 0
21 EYE_COUNTER = 0
22 WINK_COUNTER = 0
23 INPUT_MODE = False
24 EYE_CLICK = False
25 LEFT_WINK = False
26 RIGHT_WINK = False
27 SCROLL_MODE = False
28 ANCHOR_POINT = (0, 0)
29 WHITE_COLOR = (255, 255, 255)
30 YELLOW_COLOR = (0, 255, 255)
    
```

Fig: Run the code



```

207 # Show the frame
208 cv2.imshow("Frame", frame)
209 key = cv2.waitKey(1) & 0xFF
210
211 # If the 'Esc' key was pressed, break from the loop
212 if key == 27:
213     break
214
215 # Do a bit of cleanup
216 cv2.destroyAllWindows()
217 vid.release()
218
219 while True:
220     if key == 27:
    
```

Terminal: local +

Windows PowerShell

Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! <https://aka.ms/PSWindows>

PS C:\Users\DELL>

Error running 'mouse-control': Cannot run program "C:\Users\DELL\AppData\Local\Programs\Python\Python310\python.exe" (in directory "C:\Users\DELL\Desktop\Eye Controlled Mouse Cursor\Eye Controlled ... (moments as

Fig: Code Execution

SOURCE CODE:

```

from imutils import
face_utils
from utils import * import
numpy as np import pyautogui
as pag import imutils import
dlib import cv2
# Thresholds and consecutive frame length for triggering the mouse action.
MOUTH_AR_THRESH = 0.6
MOUTH_AR_CONSECUTIVE_FRAMES=15
EYE_AR_THRESH = 0.19
EYE_AR_CONSECUTIVE_FRAME= 15
WINK_AR_DIFF_THRESH = 0.04
WINK_AR_CLOSE_THRESH = 0.19
WINK_CONSECUTIVE_FRAMES = 10
# Initialize the frame counters for each action as well as
# booleans used to indicate if action is performed or not
MOUTH_COUNTER = 0
EYE_COUNTER = 0
WINK_COUNTER = 0
INPUT_MODE = False
EYE_CLICK = False
LEFT_WINK = False
RIGHT_WINK = False
SCROLL_MODE = False
ANCHOR_POINT = (0, 0)
WHITE_COLOR = (255, 255, 255)
YELLOW_COLOR = (0, 255, 255)
RED_COLOR = (0, 0, 255)
GREEN_COLOR = (0, 255, 0)
BLUE_COLOR = (255, 0, 0)
BLACK_COLOR = (0, 0, 0)
# Initialize Dlib's face detector (HOG-based) and then create
# the facial landmark predictor shape_predictor = "model/shape_predictor_68_face_landmarks.dat"
detector = dlib.get_frontal_face_detector()
predictor = dlib.shape_predictor(shape_predictor)
# Grab the indexes of the facial landmarks for the left and
# right eye, nose and mouth respectively
(lStart,lEnd)=face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
(rStart,rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]
(nStart,nEnd) = face_utils.FACIAL_LANDMARKS_IDXS["nose"]
(mStart,mEnd) = face_utils.FACIAL_LANDMARKS_IDXS["mouth"]
# Video capture vid =
cv2.VideoCapture(0) resolution_w =
cam_w = 640 cam_h = 480 unit_w =
resolution_w / cam_w unit_h =
resolution_h / cam_h while True:
# Grab the frame from the threaded video file stream, resize#
#channels )

```

```

_, frame = vid.read() frame = cv2.flip(frame, 1) frame =
# Detect faces in the grayscale frame
# Loop over the face detections
if rect = rects[0]
else: cv2.imshow("Frame" , frame)
cv2.waitKey(1) &
0xFF continue
# Determine the facial landmarks for the face region, then
# convert the facial landmark (x, y)-coordinates to a NumPy
# array shape = predictor(gray, rect) shape = face_utils.shape_to_np(shape)
# Extract the left and right eye coordinates, then use the
# coordinates to compute the eye aspect ratio for both eyes
mouth = shape[mStart:mEnd]
leftEye = shape[lStart:lEnd]
rightEye = shape[rStart:rEnd]
nose = shape[nStart:nEnd]
# Because I flipped the frame, left is right, right is left. temp = leftEye
leftEye = rightEye
rightEye = temp
# Average the mouth aspect ratio together for both
eyes mar = mouth_aspect_ratio(mouth)
leftEAR = eye_aspect_ratio(leftEye)
rightEAR = eye_aspect_ratio(rightEye)
ear = (leftEAR + rightEAR) / 2.0
diff_ear = np.abs(leftEAR - rightEAR)
nose_point = (nose[3, 0], nose[3, 1])
# Compute the convex hull for the left and right eye, then
# visualize each of the eyes mouthHull = cv2.convexHull(mouth)
leftEyeHull = cv2.convexHull(leftEye)
rightEyeHull = cv2.convexHull(rightEye)
cv2.drawContours(frame, [mouthHull], -1, YELLOW_COLOR, 1)
cv2.drawContours(frame, [leftEyeHull], -1, YELLOW_COLOR, 1)
cv2.drawContours(frame, [rightEyeHull], -1, YELLOW_COLOR, 1)
for (x, y) in np.concatenate((mouth, leftEye, rightEye), axis=0):
cv2.circle(frame, (x, y), 2, GREEN_COLOR, -1)
# Check to see if the eye aspect ratio is below the blink
# threshold, and if so, increment the blink frame counter
if diff_ear > WINK_AR_DIFF_THRESH:
if leftEAR < rightEAR
if leftEAR < EYE_AR_THRESH:
WINK_COUNTER += 1
if WINK_COUNTER > WINK_CONSECUTIVE_FRAMES: pag.click(button='left')
WINK_COUNTER = 0 elif leftEAR > rightEAR: if rightEAR < EYE_AR_THRESH:
WINK_COUNTER += 1 if WINK_COUNTER > WINK_CONSECUTIVE_FRAMES:
pag.click(button='right')
WINK_COUNTER = 0
else: WINK_COUNTER = 0
else: if ear <= EYE_AR_THRESH:
EYE_COUNTER += 1 if EYE_COUNTER > EYE_AR_CONSECUTIVE_FRAMES:
SCROLL_MODE = not SCROLL_MODE

```

```

# INPUT_MODE = not INPUT_MODE EYE_COUNTER = 0
# nose point to draw a bounding box around it else:
EYE_COUNTER = 0 WINK_COUNTER = 0
if mar > MOUTH_AR_THRESH:
MOUTH_COUNTER += 1
if MOUTH_COUNTER >= MOUTH_AR_CONSECUTIVE_FRAMES:
# if the alarm is not on, turn it on INPUT_MODE = not INPUT_MODE
# SCROLL_MODE = not SCROLL_MODE MOUTH_COUNTER = 0
ANCHOR_POINT = nose_point
else: MOUTH_COUNTER = 0
if INPUT_MODE: cv2.putText(frame, "READING INPUT!", (10, 30),
cv2.FONT_HERSHEY_SIMPLEX, 0.7, RED_COLOR, 2)
x, y = ANCHOR_POINT nx, ny = nose_point
w, h = 60, 35 multiple = 1
cv2.rectangle(frame, (x - w,y - h), (x + w, y + h), GREEN_COLOR, 2)
cv2.line(frame,
ANCHOR_POINT, nose_point, BLUE_COLOR, 2)
dir = direction(nose_point, ANCHOR_POINT, w,h)
cv2.putText(frame, dir.upper(), (10, 90),
cv2.FONT_HERSHEY_SIMPLEX, 0.7, RED_COLOR, 2)
drag = 18
if dir == 'right':
pag.moveRel(drag,0)
elif dir == 'left': pag.moveRel(drag,0)
elif dir == 'up':
if SCROLL_MODE: pag.scroll(40)

else: pag.moveRel(0, - drag)
elif dir == 'down':
if SCROLL_MODE: pag.scroll(-40)
else: pag.moveRel(0, drag)
if SCROLL_MODE:
cv2.putText(frame, 'SCROLL MODE IS ON!', (10, 60),
cv2.FONT_HERSHEY_SIMPLEX,0.7, RED_COLOR, 2)
# cv2.putText(frame, "MAR: {:.2f}".format(mar), (500, 30),
# cv2.FONT_HERSHEY_SIMPLEX, 0.7, YELLOW_COLOR,2)
# cv2.putText(frame, "Right EAR: {:.2f}".format(rightEAR),
(460, 80),cv2.FONT_HERSHEY_SIMPLEX, 0.7, YELLOW_COLOR,2)
# cv2.putText(frame, "Left EAR: {:.2f}".format(leftEAR), (460, 130),
#cv2.FONT_HERSHEY_SIMPLEX,0.7,
YELLOW_COLOR,2)
#cv2.putText(frame,"DiffEAR:{:.2f}".format(np.abs(leftEAR-rightEAR)),(460,80),
cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255),2)
cv2.imshow("Frame", frame) key = cv2.waitKey(1) & 0xFF
if key ==27: break
cv2.destroyAllWindows()
vid.release()

```

CHAPTER-5

TESTING

CHAPTER - 5

TESTING

5.1 LEVELS OF TESTING

Unit Testing

Unit Testing is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest test able part of any software. It usually has one or a few inputs and usually a single output. In procedural programming, a unit may be an individual program, function, procedure, etc. In object-oriented programming, the smallest unit is a method, which may be long to a base/super class, abstract class or derived/child class.(Some treat a module of an application as a unit. This is to be discouraged as there will probably be many individual units within that module.) Unit testing frameworks, drivers, stubs, and mock/fake objects are used to assist in unit testing.

Integration Testing

Integration Testing is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing. Some different types of integration testing are big-bang, mixed (sandwich), risky-hardest, top- down, and bottom-up. Other Integration Patterns are: collaboration integration, backbone integration, layer integration, client-server integration, distributed services integration and high- frequency integration. In the big-bang approach, most of the developed modules are coupled together to form a complete software system or major part of the system and the used for integration testing. This method is very effective for saving time in the integration testing process. However, if the test cases and their results are not recorded properly, the entire integration process will be more complicated and may prevent the testing team from achieving the goal of integration testing.

Bottom-up testing is an approach to integrated testing where the lowest level components are tested first, then used to facilitate the testing of higher-level components. The process is repeated until the component at the top of the hierarchy is tested. All the bottom or low-level modules, procedures or functions are integrated and then tested. After the integration testing of lower-level integrated modules, the next level of modules will be formed and can be used for integration testing. This approach is helpful only when all or most of the modules of the same development level are ready.

This method also help stop determine the level software developed and make site to report testing progress in the form of a percentage. Top-down testing is an approach to integrated testing where the top integrated modules are tested and the branch of the module istested step by step until the end of the related module.

System Testing

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black-box testing, and as such, should require no knowledge of the inner design of the code or logic. As a rule, system testing takes, as its input, all of the "integrated" software components that have passed integration testing and also the software system itself integrated with any applicable hardware system(s).

The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together (called assemblages) or between any of the assemblages and the hardware. System testing is a more limited type of testing; it seeks to detect defects both within the "inter-assemblages" and also within the system as a whole. System Testing (ST) is a black box testing technique performed to evaluate the complete system the system's compliance against specified requirements. In System testing, the functionalities of the system are tested from an end-to-end perspective.

System Testing is usually carried out by a team that is independent of the development team in order to measure the quality of the system unbiased. It includes both functional and Non-Functional testing.

Black Box Testing

Black-box testing is a method of software testing that examines the functionality of an application without peering into its internal structures or workings. This method of test can be applied virtually to every level of software testing: unit, integration, system and acceptance. It is sometimes referred to as specification-based testing. Specific knowledge of the application's code/internal structure and programming knowledge in general is not required. The tester is aware of what the software is supposed to do but is not aware of how it does it. For instance, the tester is aware that a particular input returns a certain, in variable output but is not aware of how the software produces the output in the first place.

White Box Testing

White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality (i.e. black-box testing). In white-box testing an internal perspective of the system, as well as programming skills. The tester chooses inputs to exercise paths through the code and determine the expected outputs. This is an analog to testing nodes in a circuit, e.g. in-circuit testing (ICT). White-box testing can be applied at the unit, integration and system levels of the software testing process. Although traditional testers tended to think of white-box testing as being done at the unit level, it is used for integration and system testing more frequently today. It can test paths within a unit, paths between units during integration, and between subsystems during a system-level test. Though this method of test design can uncover many errors or problems, it has the potential to miss unimplemented parts of the specification or missing requirements..

TEST WORK

Testing is the most popular method of assessing a framework or its components in order to see if they meet pre-determined criteria. The test involves executing a framework to identify vulnerabilities, mistakes, or a lack of requirements against what is actually required. According to ANSI/IEEE 1059, testing is the act of dismantling a product in order to distinguish between existent and required conditions (i.e. abandonment/failure/failure) and evaluating product strengths. Who conducts the tests? It is built on the interaction and collaboration of the project's partners (s). Large firms in the IT sector have a staff that evaluates the programmer generated under specific conditions. Unit tests are also carried out by designers. Experts are typically connected with testing a framework within their own personal limits:

- Programming Checker
- Programming Developer
- Team Leader / Project Manager
- End User

Different ideas can be employed when undertaking code testing at different levels. The basic levels of programming testing are:

- Check the utility
- Non-Utility Test
- Utility Test

It is a sort of black box testing that is dependent on the product's specs. The application is put to the test by giving data, and the outcomes must match the usefulness expected. The usefulness testing of a product is performed against a comprehensive integrated framework to assess the framework's consistency with its predefined prerequisites. Programmable test lifecycle.

The method involved in testing a product in an organized and deliberate manner is known as the Programmable Test Life Cycle (STLC). In any case, distinct associations have varied stages 14 in the Software Test Life Cycle (STLC) for the waterfall progression model, which incorporates accompanying phases.

- Demand analysis
- Test planning
- Test analysis
- Experimental design

Prerequisites Analysis

In this degree analyzers look into the consumer stipulations and paintings with engineers for the duration of the plan degree to peer which requirements are testable and the manner that they may check the ones stipulations.

It is essential to start trying out sports from the stipulation's degree itself in mild of the reality that the fee of solving imperfection is quite much less assuming it's far determined in requirements degree in preference to in ongoing stages.

● Test Planning

In this degree all of the education approximately trying out is carried out like what ought to be tried, how the trying out can be carried out, check technique to be followed, what's going to be the check climate, what check philosophies can be followed, system and programming accessibility, assets, probabilities and so forth. A fashionable check plan archive is made which includes all of the arranging inputs referenced above and coursed to the partners.

● Test Analysis

After check arranging degree is over check research degree begins, on this degree we really need to dig similarly into task and kind out what trying out ought to be finished in every SDLC degree. Computerization sports are moreover settled on this degree, if automation ought to be completed programming item, how would possibly the mechanization be carried out.

• Test Design

In this degree distinct black-container and white-container check plan tactics are applied to plot the experiments for trying out, analyzers start composing experiments via way of means of following the ones plan strategies, at the off threat that computerization trying out ought to be carried out, mechanization scripts moreover desires to written on this degree. 15

- The phrases ought to now no longer be rehashed with inside the game.
- The period of the phrase ought to be greater noteworthy than or equal to three.

Test Scenario ID		Load Dataset		Test Case ID		Load_Dataset-1A	
Test Case Description		Loading Facial Mark Dataset		Test Priority		High	
Pre Requisite		NA		Post Requisite		NA	
Test Execution Steps							
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result	Test Comments
1	Loading	Dataset	Dataset	Dataset		Pass	NA
	Facial Mark dataset to train the model to detect the edges of the face detected	Containing the facial land marks	Loaded	Loaded			

Table No.5.1: Loading Dataset Testcase

Test Scenario ID		Detect facial features		Test Case ID		Detect_Facial-1A	
Test Case Description		Extracting facial Contours		Test Priority		High	
Pre Requisite		NA		Post Requisite		NA	
Test Execution Steps							
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result	Test Comments
1	Detect facial features	Face Bounding Box Region	Need to extract eye, mouth, nose contours	Contours Generated		Pass	NA

Table No.5.2: Facial Features Detection Testcase

Test Scenario ID	Reading a frame			Test Case ID	Read_Frame-1A		
Test Case Description	Read a fram from the video stream			Test Priority	High		
Pre Requisite	NA			Post Requisite	NA		
Test Execution Steps							
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result	Test Comments
1	Detect face	Gray scale image	Need to extract face region from the given gray scale image	Face Detected successfully		Pass	NA

Table No.5.3: Face Detection Testcase

Test Scenario ID		Mouse Operations		Test Case ID		Mouse_Operations-1A	
Test Case Description		Checking the working of mouse operation		Test Priority		High	
Pre Requisite		NA		Post Requisite		NA	
Test Execution Steps							
S.No	Action	Inputs	Expected Output	Actual Output	Test Browser	Test Result	Test Comments
1	Mouse cursor movement	Facial moment	Cursor movement	Cursor moved		Pass	NA
2	Left and right Clicking operations	Left and Right Eye Winking	Left and Right Click operations	Left and right clicks worked		Pass	

Table No.5.4: Mouse Clicking Operations Test Cases

CHAPTER-6

SCREENSHOTS

CHAPTER - 6

SCREENSHOTS

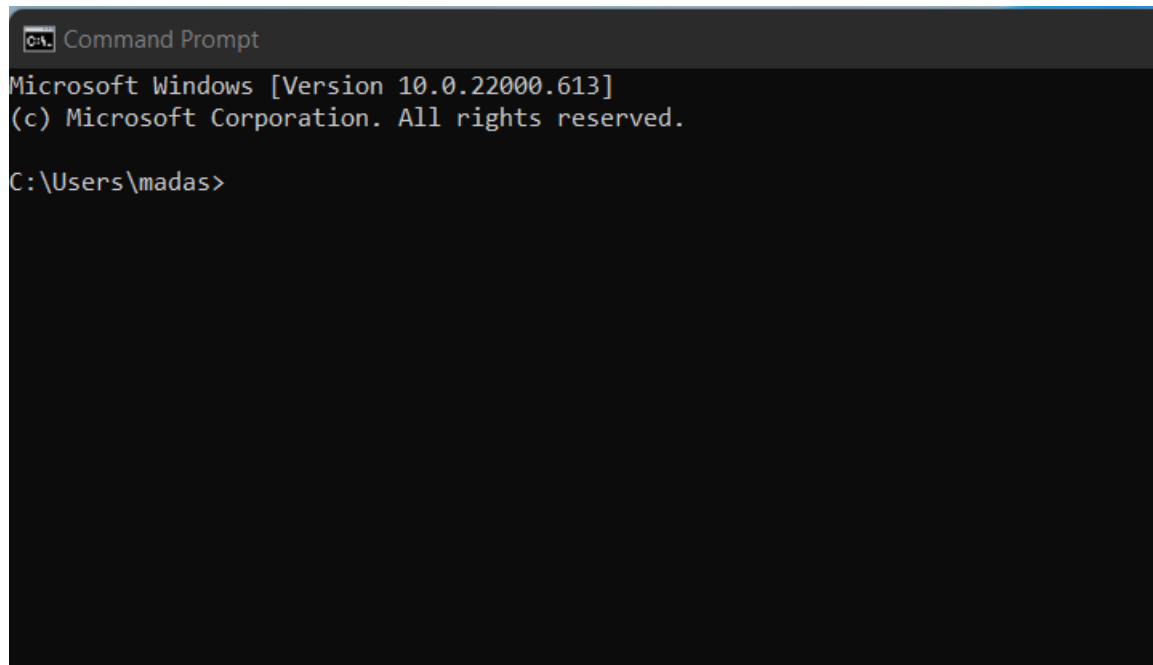


fig.15: Open the Command Prompt

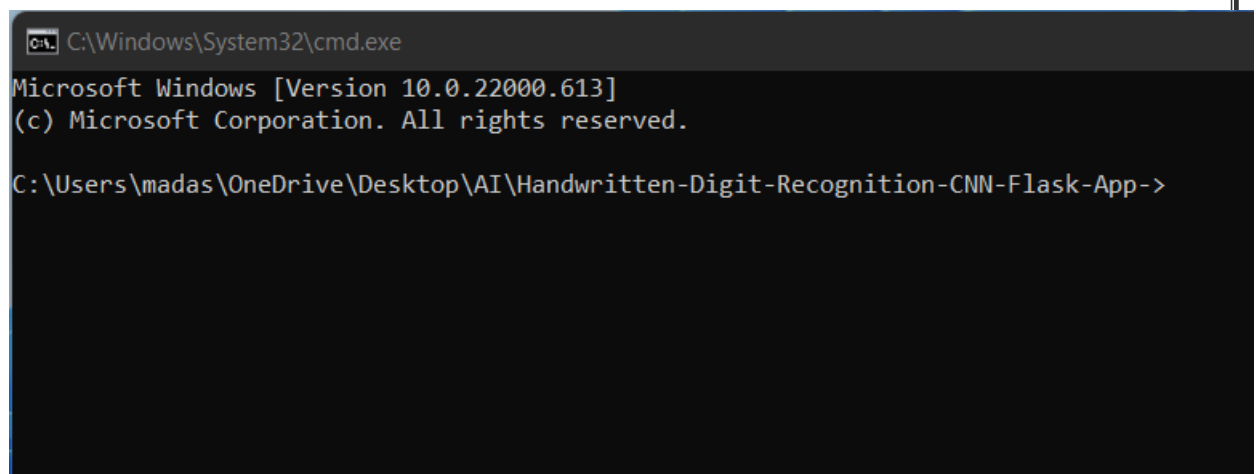


fig16: Go to the directory where the application is present

```
C:\Windows\System32\cmd.exe - py app.py
Microsoft Windows [Version 10.0.22000.613]
(c) Microsoft Corporation. All rights reserved.

C:\Users\madras\OneDrive\Desktop\AI\Handwritten-Digit-Recognition-CNN-Flask-App->py app.py
* Serving Flask app 'app' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Restarting with stat
* Debugger is active!
* Debugger PIN: 237-651-676
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Fig 17: Run the module

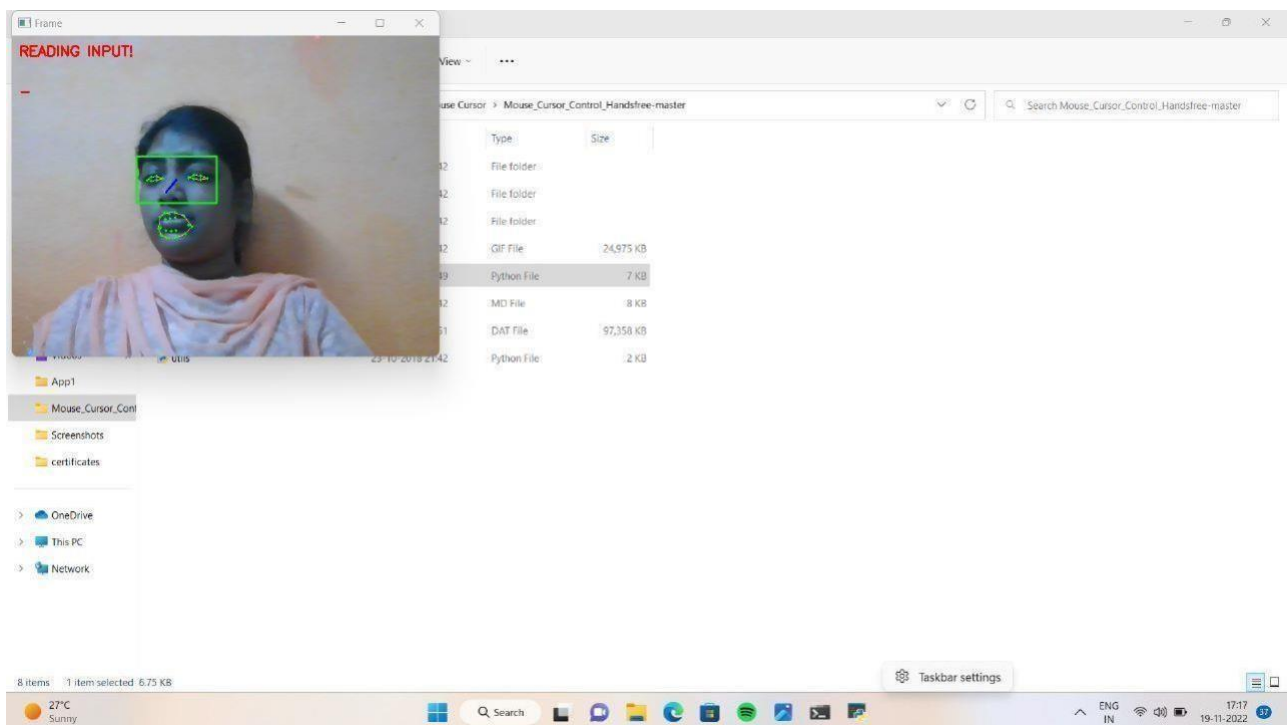


fig. 6.1 Reading Input

SAMPLEINPUTSANDOUTPUTS

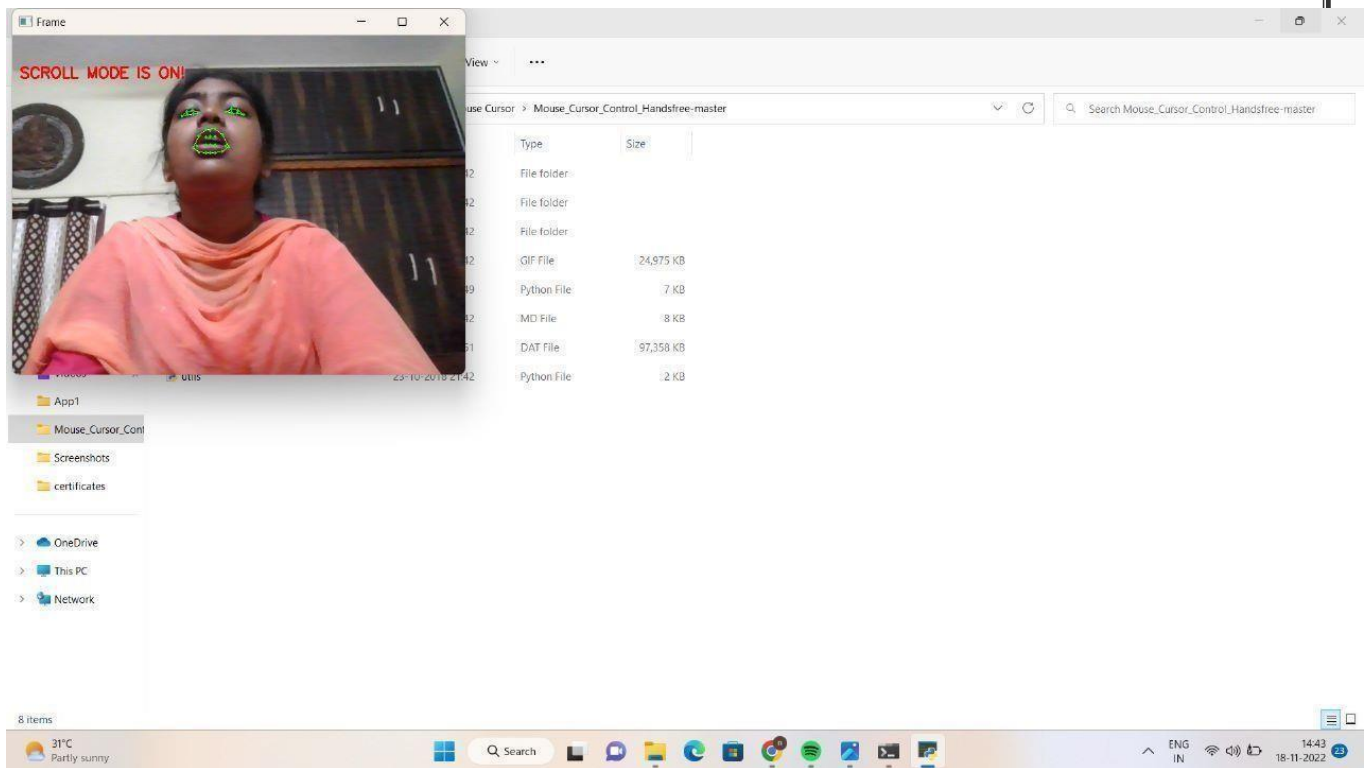


fig.6.2: Scroll mode on

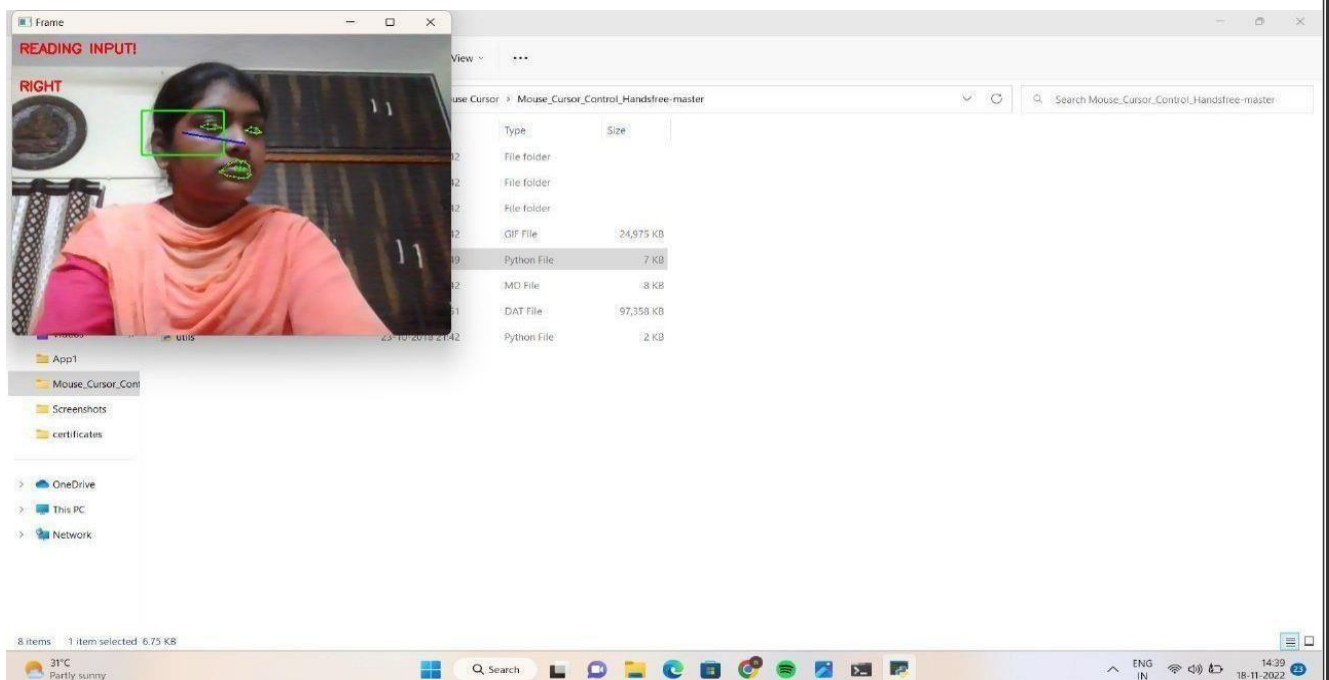


Fig.6.3: Scrolling Right

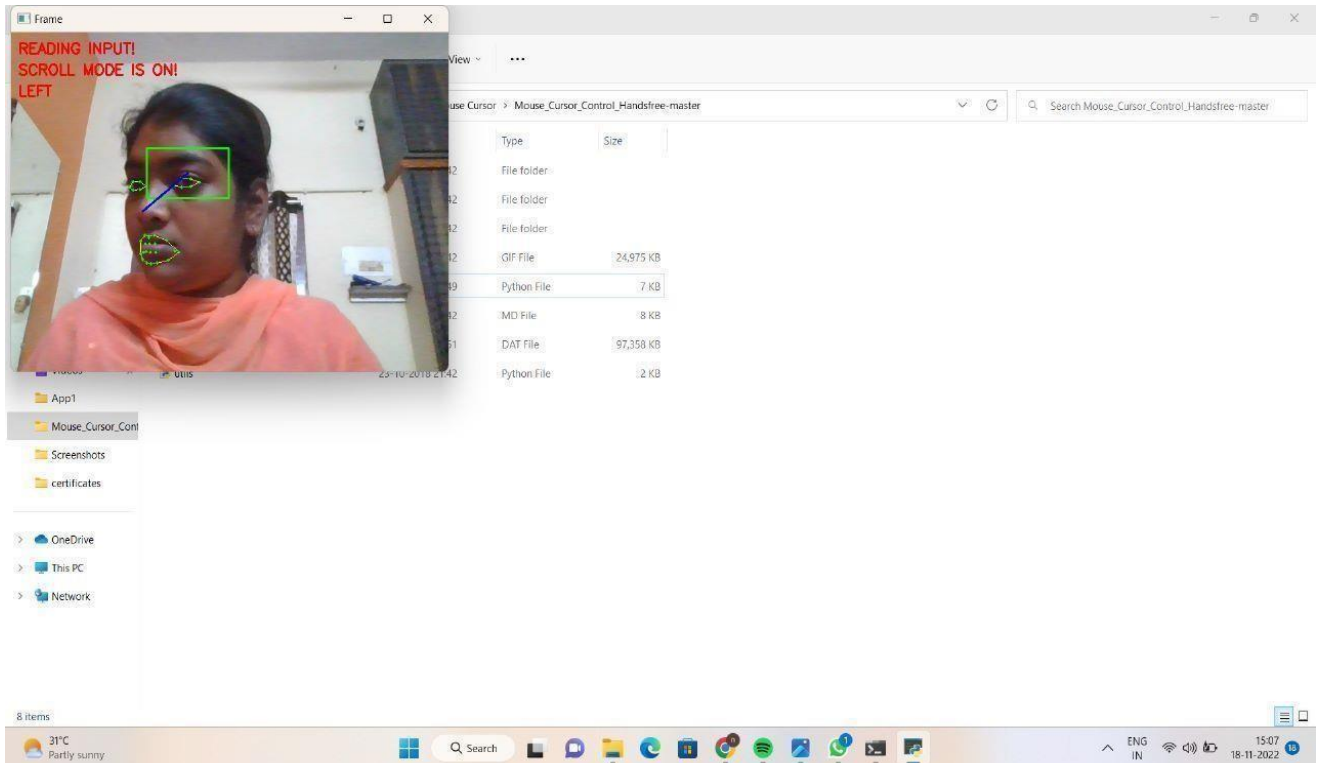


Fig. 6.4: Scrolling Left

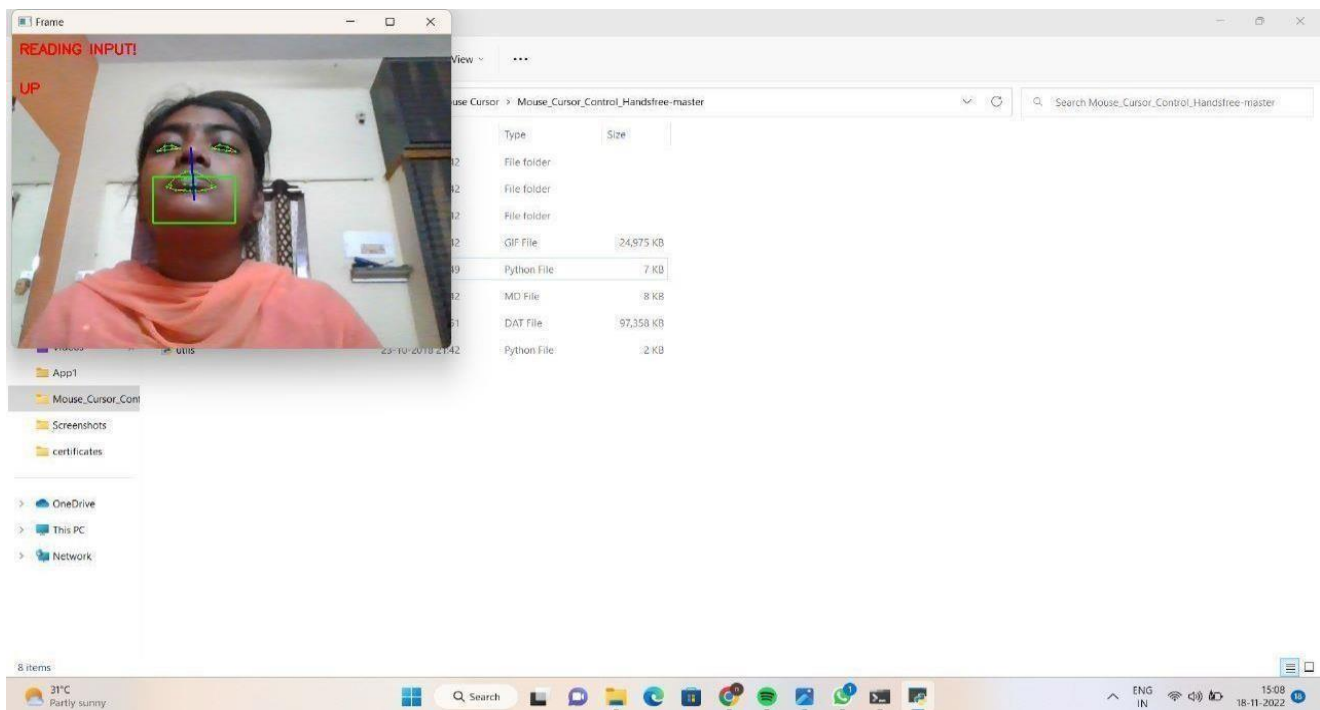


Fig. 6.5: Scrolling Up

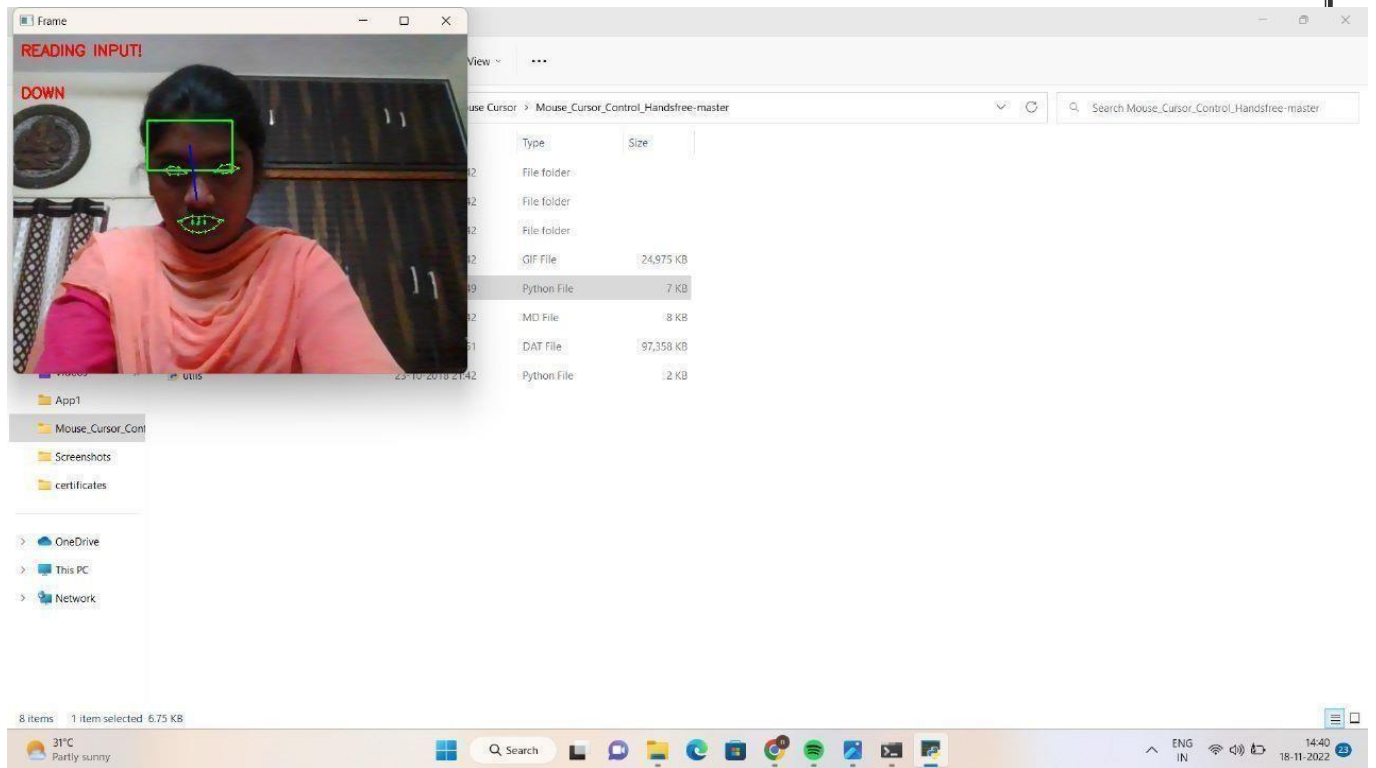


Fig. 6.6: Scrolling Down

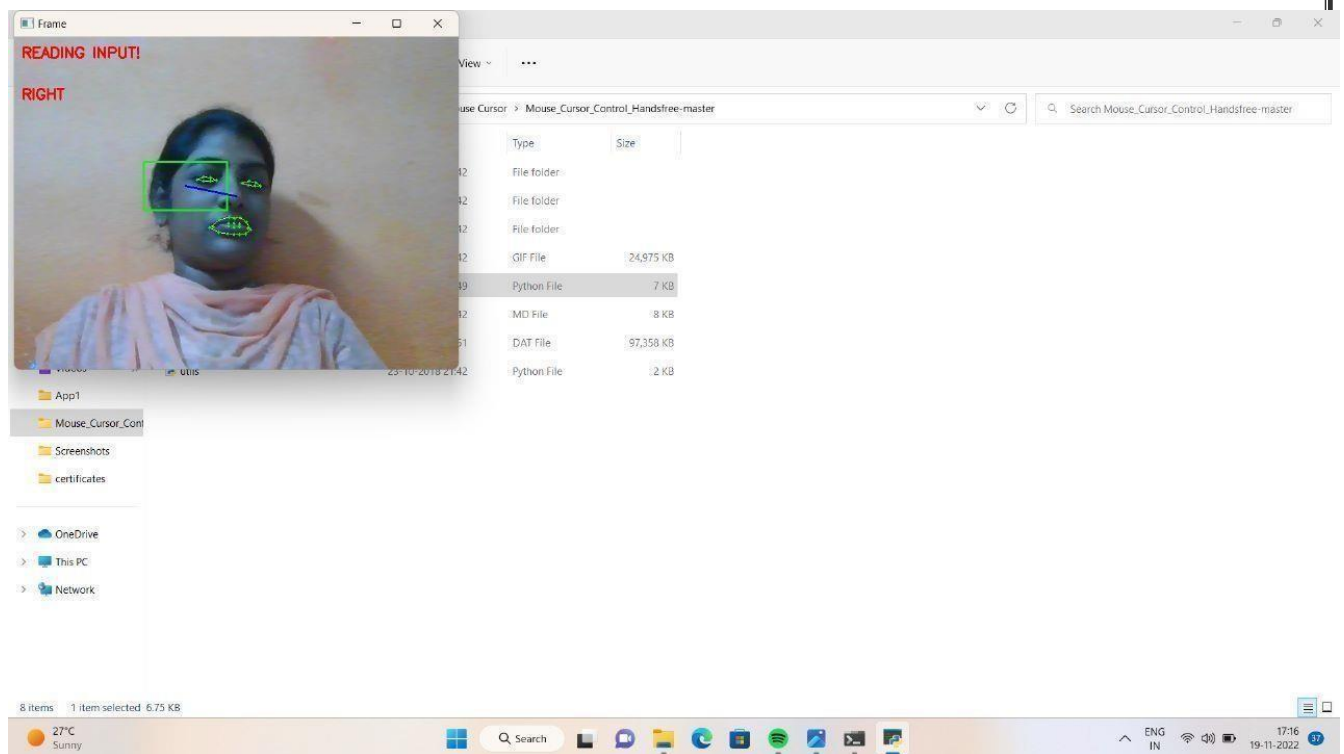


Fig. 6.7: Right movement of cursor

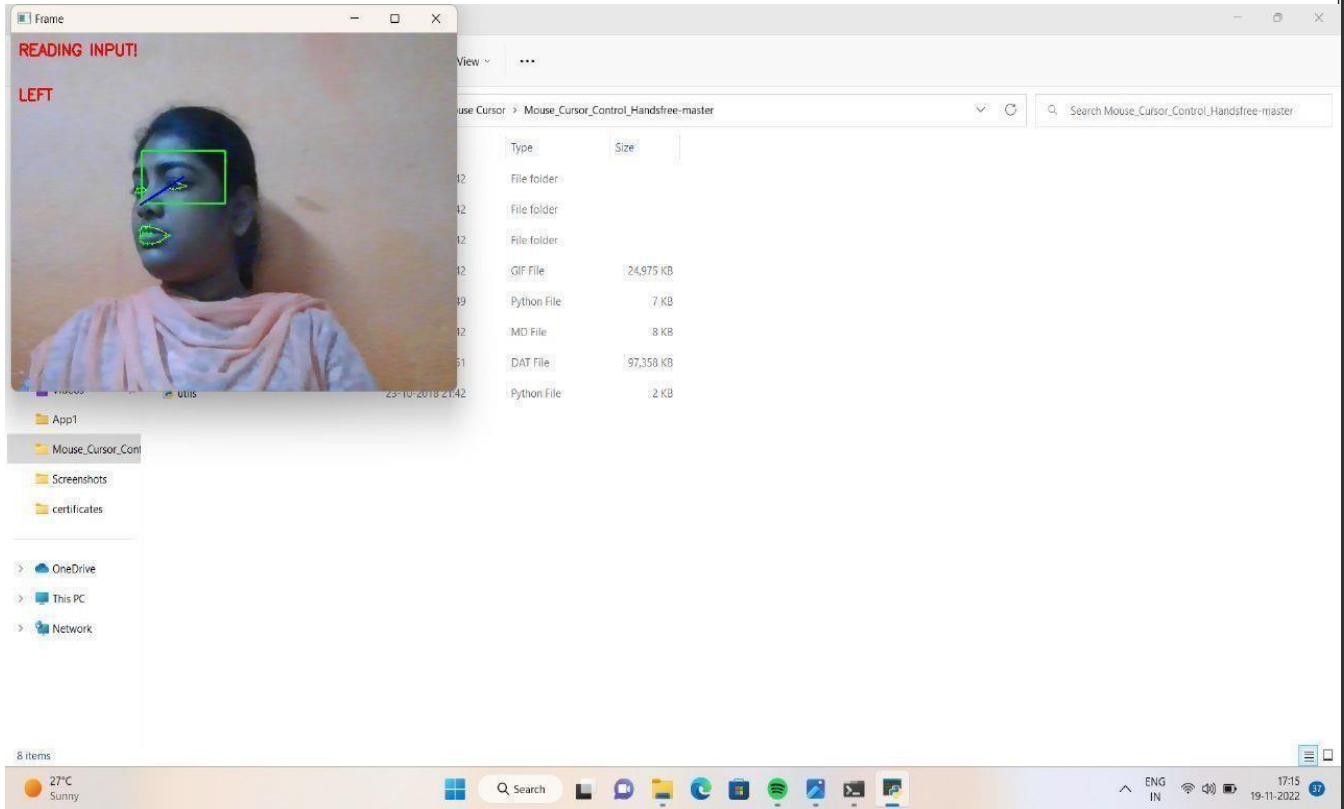


Fig. 6.8: Left movement of cursor

CHAPTER-7

CONCLUSION

CHAPTER-7

CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

From the process implemented it is cleared that the cursor can be controlled by the eyeball movement i.e., without using hands on the computers. This will be helpful for the people having disability in using the physical parts of the computers to control the cursor points. Because the cursor points can be operated by moving the eyeballs. Without the help of others disabled people can use the computers. This technology can be enhanced in the future by inventing more techniques like clicking events as well as to do all the mouse movements and also for human interface systems using eye blinks. Technology also extended to the eyeball movement and eye blinking to get the efficient and accurate movement.

7.2 FUTURE SCOPE

In order to make user interact with computer naturally and conveniently by only using their eye, we provide an eye tracking based control system. The system not only enables the disabled users to operate the computer the same as the normal users do but also provides normal users with a novel choice to operate computer. Currently, this system is applied for the general operating behaviour to interact with computer by simulating mouse. In future, we can add new operation functions for more usage situations for users to communicate with media and adjust our system on new platform, such as tablet or phone.

REFERENCES

REFERENCES

- D. G. Evans, R. Drew, and P. Blenkhorn, "Controlling mouse pointer position using an infrared head-operated joystick," 2000.
- M. Betke, J. Gips, and P. Fleming, "The Camera Mouse: Visual Tracking of Body Features to Provide Computer Access for People With Severe Disabilities," On Neural Systems And Rehabilitation Engineering, March 2002.
- M. Nabati and A. Behrad, "Camera Mouse Implementation Using 3D Head Pose Estimation by Monocular Video Camera and 2D to 3D Point and Line Correspondences," 2010 5th International Symposium on Telecommunications (IST'2010), 2010.
- Y. L. Chen, F. T. Tang, W. H. Chang, M. K. Wong, Y. Y. Shih, and T.S. Kuo, "The new design of an infrared controlled human-computer interface for the disabled," Dec. 1999.
- Sivasangari, A. Deepa, D. Anandhi, T. Anitha ponjari and Roobina. M. S Eyeball based cursor Movement Control International Conference on Communication and Signal Processing, July 28-30-2020, India.
- J. Sreedevi, M. Shreya Reddy, B. Satyanarayana Eyeball Movement based cursor using Raspberry pi and OpenCV International journal of innovative technology and exploring engineering (IJTEE) ISSN, volume-9 issue-7, May 2020.
- R. Rithil, V. Manjusri, M. Yeshodha, G. Renuka cursor control using eyeball movement using raspberry pi.
- Kollipara sai varun, Puneeth, Dr. T. Prem Jacob, Virtual mouse implementation using OpenCV proceedings of the third international conference on trends in electronics and information.
- Alax Poole and Linden J. Ball, Eye Tracking in human-computer interaction and usability research: current status and future prospects ,in Encyclopedia of human computer interaction (30 December 2005).

MAPPING OF CO'S AND PO'S

Program Outcomes (Pos):

Engineering Graduates will be able to:

1. Engineering knowledge:

Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. Problem analysis:

Identify, formulae, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural science, and engineering science.

3. Design/development of solution:

Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4. Conduct investigations of complex problems:

Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions, component, or software to meet the desired needs.

5. Modern tool usage:

Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.

6. The engineer and society:

Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. Environmental and sustainability:

Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need of sustainable development.

8. Ethics:

Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. Individual and team work:

Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. Communication:

Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. Project management and finance:

Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. Life-long learning:

Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Specific Outcomes (PSOs):

PSO1: Design, develop, test and maintain reliable software systems and intelligent systems.

PSO2: Design and develop websites, web apps and mobile apps.

PROJECT PERFORMANCE

Classification of Project	Application	Product	Research	Review
	✓			

Project Outcomes

Course Outcome (CO1)	Identify and analyze the problem statement using prior technical knowledge in the domain of interest.
Course Outcome (CO2)	Design and develop engineering solutions to complex problems by employing systematic approach.
Course Outcome (CO3)	Examine ethical, environmental, legal and security issues during project implementation.
Course Outcome (CO4)	Prepare and present technical reports by utilizing different visualization tools and evaluation metrics.

Mapping Table

IT2522: MINI PROJECT

Course Outcomes	Program Outcomes and Program Specific Outcomes														
	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12		PSO 1	PSO 2
CO1	3	3	1					2	2	2				1	1
CO2	3	3	3	3	3			2	2	2		1		3	3
CO3	2	2	3	2	2	3	3	3	2	2	2			3	
CO4	2		1		3				3	3	2	2		2	2