A

Mini Project Report

On

HUMAN COMPUTER INTERACTION BASED EYE CONTROLLED MOUSE

Submitted in partial fulfillmentof the requirement for the Award of the Degree in

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE & ENGINEERING

Submitted by

D. Manichandan 1608-20-733-155

Under the guidance of

Mrs.M.Priyanka

Asst.professor (CSE Dept.)

Department of Computer Science and Engineering

Matrusri Engineering College

Accredited by NBA & NAAC

(Affiliated to Osmania University, Approved by AICTE)

Saidabad, Hyderabad-500059

2022-2023

Department of Computer Science and Engineering

Matrusri Engineering College

Accredited by NBA & NAAC

(Affiliated to Osmania University, Approved by AICTE)

Saidabad, Hyderabad-500059

2022-2023

CERTIFICATE

This is to certify that the Project Report entitled **"HUMAN COMPUTER INTERACTION BASED EYE CONTROLLED MOUSE"** is being submitted by D. Manichandan (1608-20-733-155) in partial fulfillment of the requirements for the award of the Degree of **Bachelor of Engineering** in **"Computer Science & Engineering"** O.U., Hyderabad during the year 2022-2023 is a record of bonafide work carried out by them under my guidance. The results presented in this Project Work have been verified and are found to be satisfactory.

Project Guide	**Head of the Department**
Mrs. M. Priyanka	**Dr. P. Vijayapal Reddy**
Asst.Professor	**Professor & HOD**
Dept. of CSE	**Dept. of CSE**

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to our advisor, **Mrs. M. Priyanka, Asst.Professor, CSE Dept.,** whose knowledge and guidance has motivated us to achieve goals we never thought possible. The time we have spent working under her/his supervision hastruly been a pleasure.

The experience from this kind of work is great and will be useful to us in future. We thank **Mrs. M. Priyanka, Asst.Professor** CSE Dept**.** for her effort, kind cooperation, guidance and encouraging us to do this work and also for providing the facilities to carryout this work.

I express my sincere thanks to Mini Project Coordinator **Dr. L. k. Indumathi, Mrs. J. Samatha** Department of Computer Science and Engineering, Matrusri Engineering College, for their valuable suggestions and constant helping in completing the work.

I express my sincere gratitude to **Dr. P. Vijayapal Reddy**, Professor & Head of the Department of Computer Science and Engineering, Matrusri Engineering College, for his precious suggestions, motivation and co-operation.

I express my sincere thanks to **Dr. D. Hanumanth Rao**, Principal, Matrusri Engineering College, Saidabad, Hyderabad, for his encouragement and constant help.

I extend my sincere thanks to all the teaching and non-teaching staff of Computer Science and Engineering Department for their support and encouragement.

Last but not least, I wish to acknowledge my friends and family members for giving moral strength moral strength and helping us to complete this dissertation.

DECLARATION

I hereby declare that the work which is being presented in this dissertation entitled, "**Human computer interaction based eye controlled mouse**", submitted towards the partial fulfillment of the requirements for the award of the degree of **Bachelor of Engineering** in **Computer Science & Engineering, Matrusri engineering college**, is an authentic record of my own work carried out under the supervision of **Mrs. M. Priyanka, Asst.Professor, Department of CSE,** Matrusri Engineering college, Saidabad.

To the best of our knowledge and belief, this project bears no resemblance with any report submitted to OU or any other University for the award of any degree or diploma.

D. Manichandan (1608-20-733-155)
Dept of CSE, MECS

Date:

Place: Hyderabad

ABSTRACT

Eye Movement can be regarded as a pivot real-time input medium for human computer communication, which is important for people with physical disability. The proposed system focuses on providing a simple and convenient interactive mode by only using user's eye. The system built is an eye-based interface that acts as a computer mouse to translate eye movements such as blinking, gazing, and squinting towards the mouse cursor actions. In this project two interactive tasks with different difficulty were done to compare the proposed eye control tool with an existing system. It presents a HCI system that is great important to amputees and those who have issues with using their hands. The system in discussion makes use of simple webcam and its software requirements are Python, OpenCV and a few other packages. An algorithm to carry out the functions of a mouse by providing a hand free interaction between humans and computers by using different expression of a face using computer vision and matching it with already stored expression. The "Camera Mouse" system tracks the computer user's movements with a video camera and translates them into the mouse movements of the mouse pointer on the screen.

Table of Contents

1. INTRODUCTION

Eye tracking technology, which is based on an eye tracker that measures the movement and positions of the eye has played an increasingly important role in psychology, marketing , and user interfaces. Eye trackers have existed for a number of years, but, early in the development of the field of eye tracking, the use of eye trackers was largely confined to laboratory experiments to observe the nature of human eye movements, rather than to use these movements as an actual control medium within a human-computer interaction (HCI) . Because the cost of eye trackers was around 30,000 a decade ago, it was too expensive to consider use in real user-computer interfaces. In recent years, with the development of better and cheaper components for gaze interaction, low-cost eye trackers have been produced by several high-profile companies, such as Tobii's EyeX tracker , GazePoint's GP3 tracker , and the Eye Tribe Tracker . As eye tracking gear gets cheaper, new applications with the concept of using eye tracking in HCI are clearly beginning to blossom.

Traditional user interfaces provide much more bandwidth from computer to user, such as images, animations, videos, and other media which can output large amounts of information rapidly. Whereas there are hardly any means of inputting comparably large amounts of information from users. The concept of HCI is to increase the bandwidth from user to computer with more natural and more convenient communication mechanisms. The eye is one of our mainly input mediums, and about 80 to 90 percent of the outside world information is obtained from the human eye . For multimedia communication from user to computer, the eye movements can be regarded as a pivotal real-time input medium, which is especially important for people with motor disability (such as persons with Amyotrophic Lateral Sclerosis) . The research of eye tracking technique in user-computer dialogue is mainly focused on incorporating eye movements into the multimedia communication with computer in a convenient and natural way.

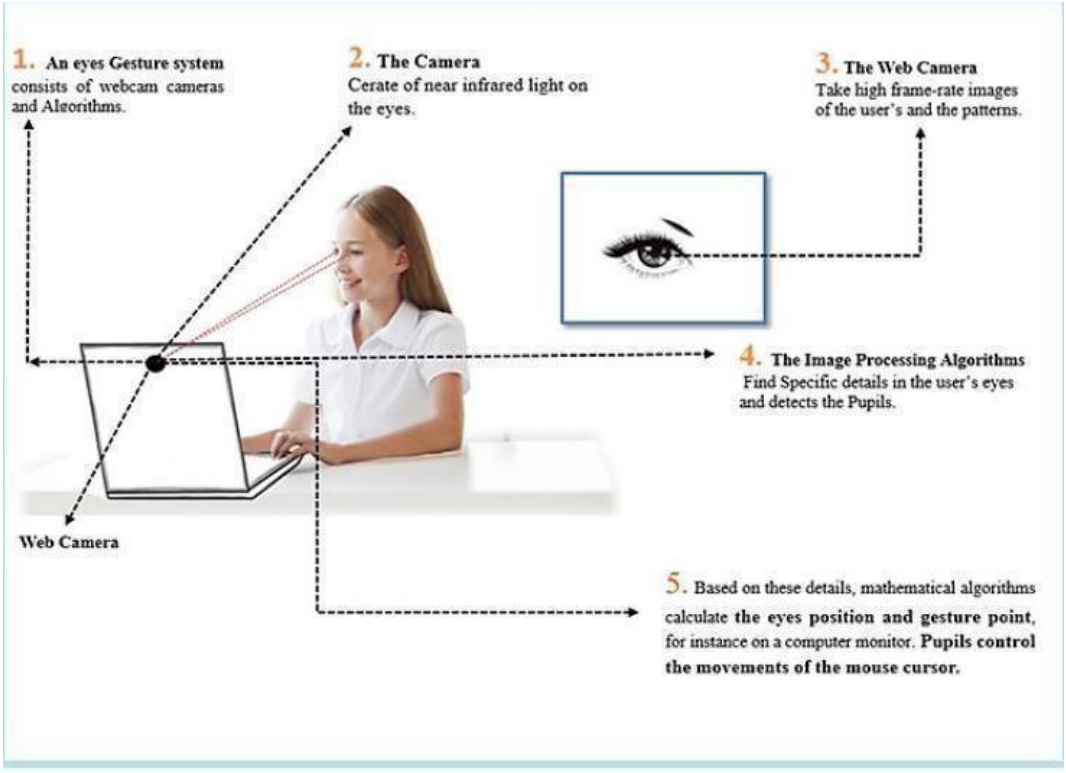
Generally, the most intuitive solution for incorporating eye movements into usercomputer dialogue would be substituted with an eye tracker directly for a manual input source, such as a mouse. By installing an eye tracker and using its , coordinate output stream as a virtual mouse, the moving of user's gaze would directly cause the mouse cursor to move. But the natural hand moving a mouse is very different from the eye movement to control virtual mouse. There are significant differences between the mouse and eye position to be considered in designing eye tracking based control system

for user-computer dialogue.

In order to provide appropriate communication, several improved eye tracking based control systems were developed.

Although the eye control systems mentioned above are of benefit for users with physical or cognitive handicaps to interact with computers appropriately, the designed eye trackers are quite complicated and expensive. Users should wear inconvenient devices and make specific actions to control the system. To lower the threshold of usability for user, MastaLomaster developed a prototype of eye control system that is based on low-cost eye trackers. The system supports most commercial low-cost eye trackers, such as Tobii EyeX and Eye Tribe. However, user should choose the desired function first and then do the real interaction with computer, which goes against user intuition and it is not natural to use.

In order to provide more natural and more convenient communication mechanisms, we present an eye tracking based control system for user-computer dialogue in this paper. The system combines both the mouse functions and keyboard functions, so that users can use our system to achieve almost all of the inputs to the computer without traditional input equipment. It was oriented towards improving the reliability, mobility, and usability for user to interact with computer by only using their eyes.



2. LITERATURE SURVEY

The system consists of a standard electric wheelchair with an on-board computer, sensors and a graphic user interface run by the computer. On the other hand, this eye-control method can be applied to handle graphical interfaces, where the eye is used as a mouse computer. Results obtained show that this control technique could be useful in multiple applications, such as mobility and communication aid for handicapped persons.

The aim and scope of the journal is to emphasize research, development and application within the fields of Scientific Research Engineering &Technology that support high-level of learning, teaching, development and research. It is an international journal that aims to contribute to the constant research and training to promote research in the relevant field.

If the user sees the monitor, the center of a pupil is always in a polygon that is made by the glints. Consequently, the direction of the user's eye gaze can be computed without computing the geometrical relation between the eye, the camera and the monitor in 3D space. Our method is comparatively simple and fast. We introduce the method and show some experimental results.

3.Analysis

3.1 Existing System:

Currently eye tracking mouse technology is not available at a large scale. The computer mouse or moving the finger has been a very common approach to move the cursor along the screen in the current technology.

Limitations:

It does not work with few users who wear contact lenses or have long eye lashes.

It requires some calibration time before it gives satisfactory results. Hence few users deviate themselves from using it.

Eye movements of some users are often un-intentional. This results into unwanted responses by the system.

It is difficult to control eye position accurately all the times unlike mouse. Eye tracker provides instable output when it does not get appropriate image of the eye

3.2 Proposed System:

The system proposed in this works based on the following action:

- Squinting your eyes
- Winking
- Blinking
- Gazing

4. Design

The database tables are designed by analyzing functions involved in the system and format of the fields is also designed. The fields in the database tables should define their role in the system. The unnecessary fields should be avoided because it affects the storage areas of the system. Then in the input and output screen design, the design should be made user friendly.

The menu should be precise and compact.

4.1 System Architecture:

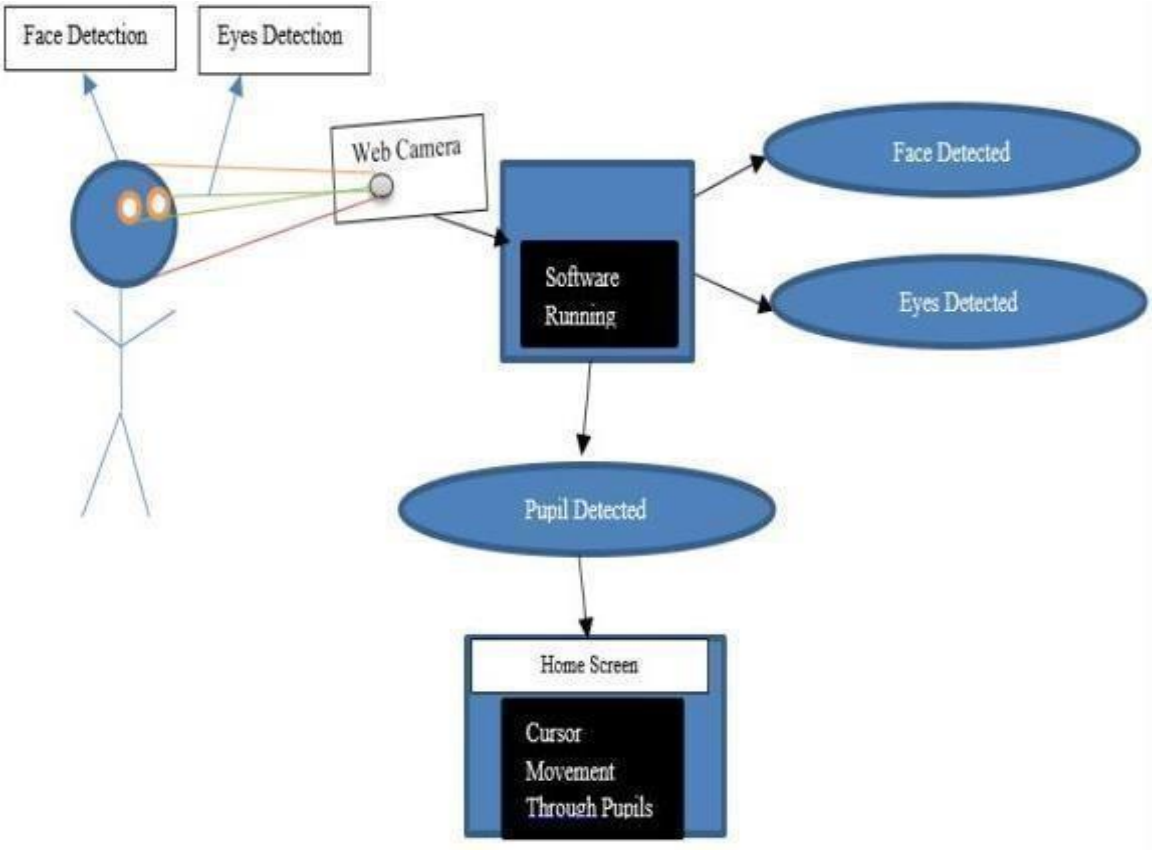


Fig 4.1: System Architecture

4.2 Data Flow Diagram:

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.

2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. Data flow diagrams are used to graphically represent the flow of data in a business information system. DFD describes the processes that are involved in a system to transfer data from the input to the file storage and reports generation.
4. Data flow diagrams can be divided into logical and physical. The logical data flow diagram describes flow of data through a system to perform certain functionality of a business. The physical data flow diagram describes the implementation of the logical data flow.

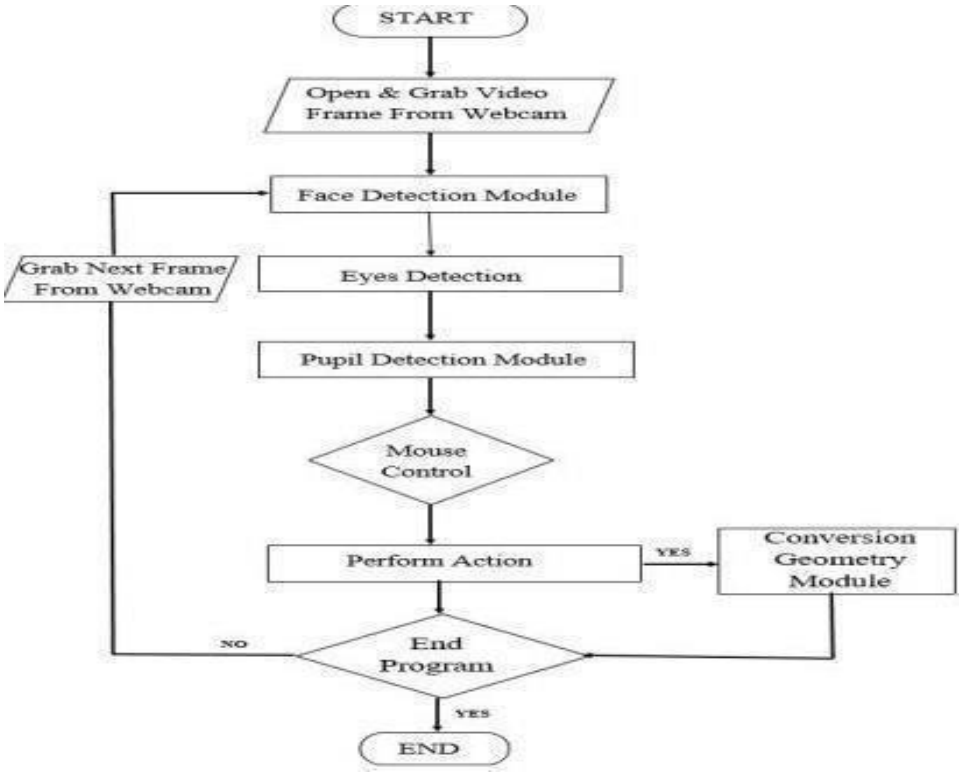


Fig 4.2: Data Flow Diagram

4.3 UML Diagrams:

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a

notation.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

Goals

The Primary goals in the design of the UML are as follows:

- Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
- Provide extendibility and specialization mechanisms to extend the core concepts.
- Be independent of particular programming languages and development process.
- Provide a formal basis for understanding the modeling language.
- Support higher level development concepts such as collaborations, frameworks, patterns and components.

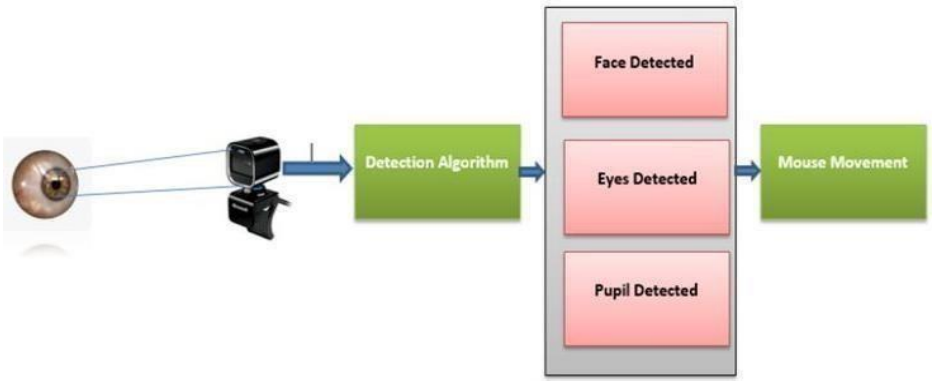


Fig Detecting the facial expressions

Use case diagram

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

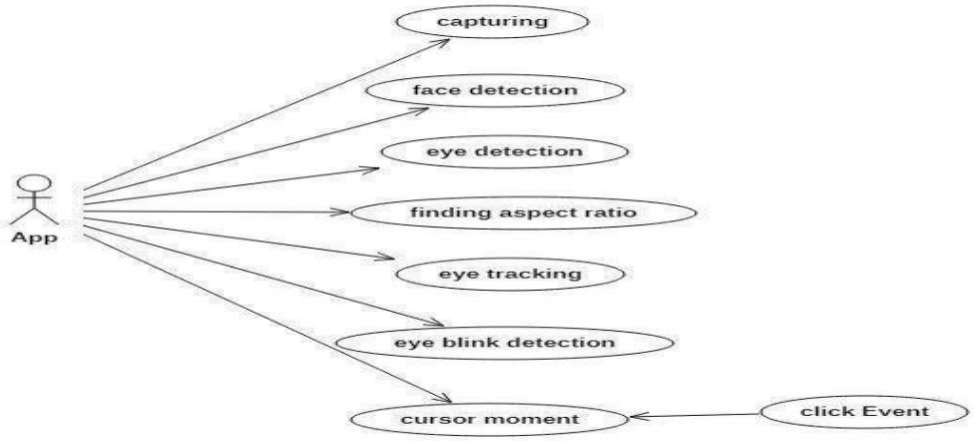


Fig 4.3.1 Use case Diagram for cursor movements

Class Diagram

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

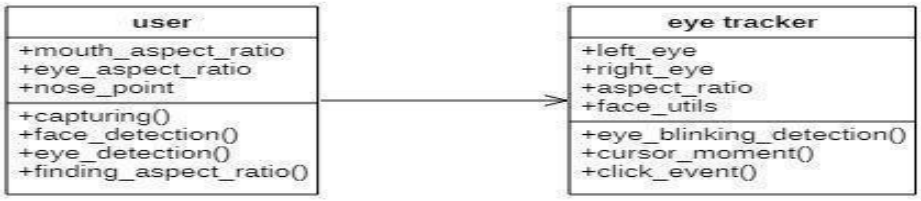


Fig 4.3.2: Class Diagram for Cursor Movements

Activity Diagram

Activity diagrams are graphical representations of Workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

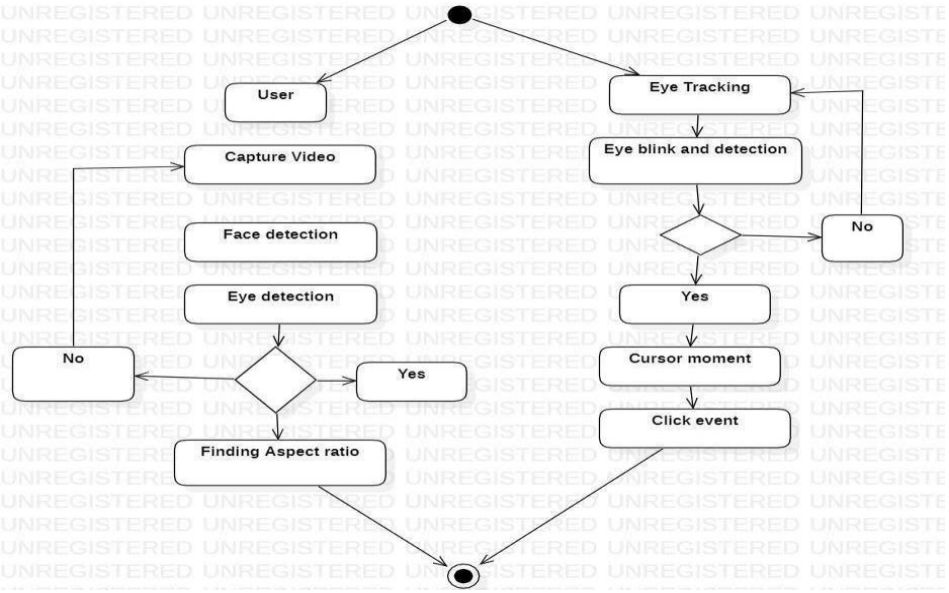


Fig 4.3.3: Activity Diagram for Cursor Movements

Sequence Diagram

Sequence Diagrams Represent the objects participating the interaction horizontally and time vertically.

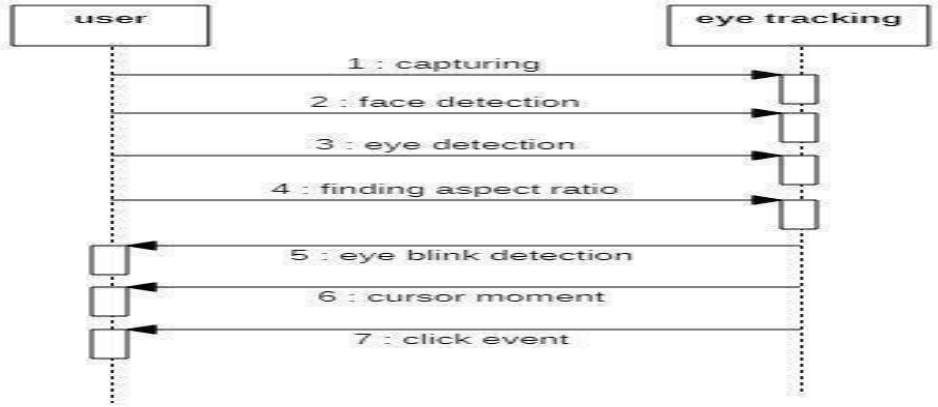


Fig 4.3.4: Sequence Diagram for Cursor Movements

State Chart Diagram

State chart diagram is one of the five UML diagrams used to model the dynamic nature of a system. They define different states of an object during its lifetime and these states are changed by events. State chart diagrams are useful to model the reactive systems. Reactive systems can be defined as a system that responds to external or internal events.

State chart diagram describes the flow of control from one state to another state. States are defined as a condition in which an object exists and it changes when some event is triggered. The most important

purpose of State chart diagram is to model lifetime of an object from creation to termination. State chart diagrams are also used for forward and reverse engineering of a system.

Following are the main purposes of using State chart diagrams −

- To model the dynamic aspect of a system.
- To model the life time of a reactive system. To describe different states of an object during its life time. Define a state machine to model the states of an object.
- To describe different states of an object during its life time. Define a state machine to model the states of an object.

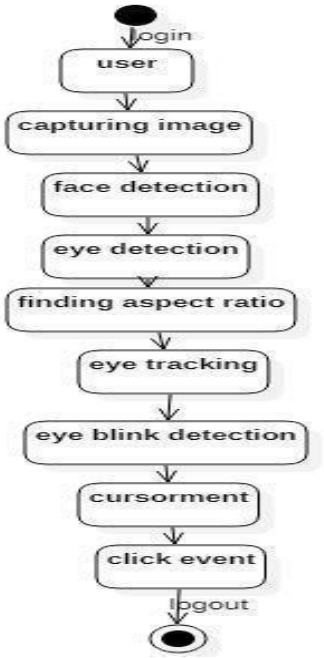


Fig 4.3.5: State Chart Diagram for Cursor Movements

5.Implementation

Currently, for face detection, perhaps deep learning models perform the best. But face detection was there before the emergence of deep learning as well. Earlier, classical feature descriptors and linear classifiers were a really good solution for face detection. And the mediapipe library provides one such classical solution for face detection. That is, HOG and Linear SVM. This is based on the **HOG** (Histogram of Oriented Gradients) feature descriptor with a **linear SVM** machine learning algorithm to perform face detection.

HOG is a simple and powerful feature descriptor. It is not only used for face detection but also it is widely used for object detection like cars, pets, and fruits. HOG is robust for object detection because object shape is characterized using the local intensity gradient distribution and edge direction.

Step 1: The basic idea of HOG is dividing the image into small connected cells.

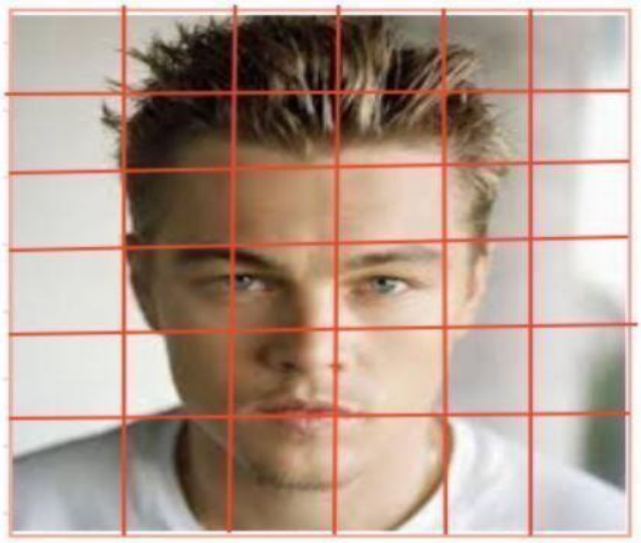


Fig 5.1:image into small connected cells

Step 2: Computes histogram for each cell**.**

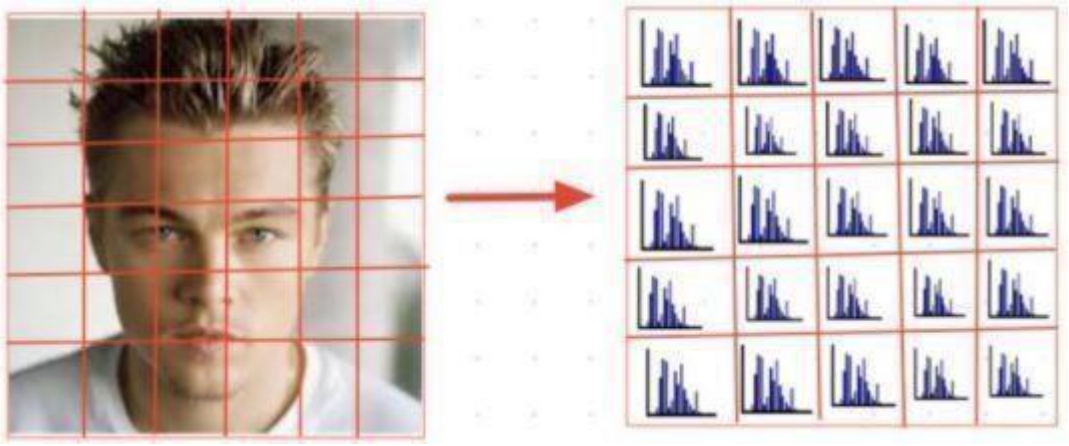


Fig 5.2: histogram for each cell

Step 3: Bring all histograms together to form feature vector i.e., it forms one histogram from all small histograms which is unique for each face.

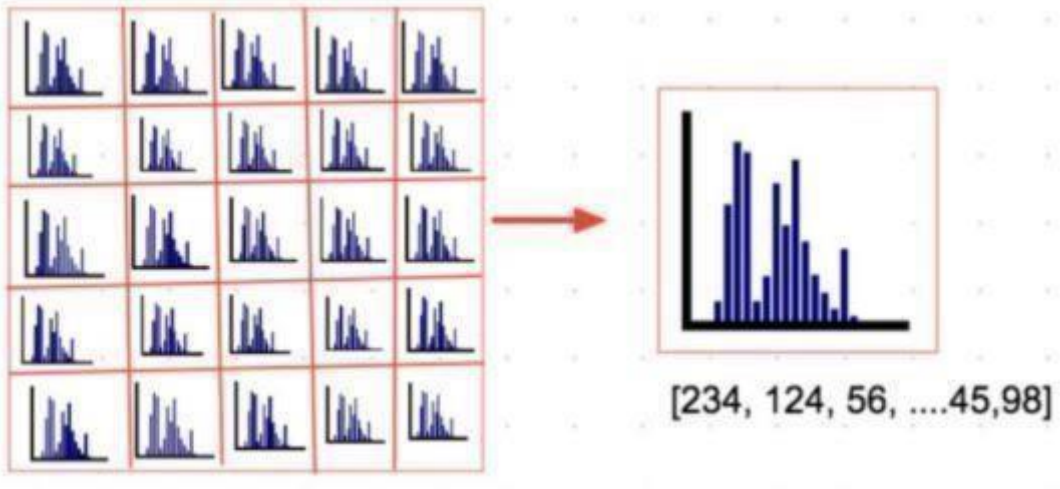


Fig 5.3: forming unique histogram

5.1 Detection of Actions Performed by the face:

After the ratios are defined, the frame can now compare the ratios of the parts of the face with the ratios defined for different actions, of the current frame being processed.

It is done using the 'if' statement. The actions which the program identifies are:

i. **For activating the Mouse:** The user needs to 'yaw' which is opening his mouth,

vertically, in turn increasing the distance between the corresponding 2D points of the mouth. The algorithm detects the change in the distance by computing the ratio, and when this ratio crosses a specified threshold, the system is activated and the cursor can be moved. The user needs to place his nose towards, either the top, bottom, left or right of a rectangle that appears, to move the cursor in the corresponding direction. The more he is away from the rectangle, the faster is the movement of the cursor.

ii. **Left/Right Clicking:** For clicking, he needs to close any one of his eyes, and make sure to keep the other open. The program first checks whether the magnitude of the difference is greater than the prescribed threshold by using the difference between the ratios of the two eyes, to make sure that the user wants to perform either the left or right click, and does not want to scroll(For which both the eyes need to squint).

iii. **Scrolling:** The user can scroll the mouse, either upwards or downwards. He needs to squint his eyes in such a way that the aspect ratio of both the eyes is less than the prescribed value. In this case, when the user places his nose outside the rectangle, the mouse performs scroll function, rather than moving the cursor. He can move his nose either above the rectangle to scroll upwards, or move it below the rectangle to scroll downwards.

6.Testing

6.1 Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.
- To ensure that during operation the system will perform as per specification.
- To make sure that system meets the user requirements during operation
- To make sure that during the operation, incorrect input, processing and output will be detected
- To see that when correct inputs are fed to the system the outputs are correct
- To verify that the controls incorporated in the same system as intended
- Testing is a process of executing a program with the intent of finding an error
- A good test case is one that has a high probability of finding an as yet undiscovered error

6.2 Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct pag**e**

The actual purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner.

6.3 TYPES OF TESTING

- Unit testing.
- Black box testing.
- White box testing.
- Integration testing.
- Functional Testing.
- System testing.

There are many types of testing methods are available in that mainly used testing methods are as follows

6.3.1 Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

6.3.2Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

6.3.3 Functional testing

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.
Functional testing is centered on the following items:

Valid Input: identified classes of valid input must be accepted.

Invalid Input: identified classes of invalid input must be rejected.

Functions: identified functions must be exercised.

Output: identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

6.3.4 System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

6.3.5 White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

6.3.6 Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

6.3.7 Test Cases

Tested	**Test name**	**Inputs**	**Expected output**	**Actual Output**	**status**

1	Load Dataset	Facial Land Mark Dataset	Read dataset	Load dataset	success
2	Capture Frame from Camera	Video Stream	Need to read frame from Camera	Reading Frame is done	success
3	Detect Face	Gray Scale Image	Need to Extract ace Region from Given Gray Scale Image	Face Detected Successfully	success
4	Detect Facial Features	Face Bounding Box Region	Need to Extract Eye Mouse Nose Contours	Contours Generated	success
5	Calculate Aspect Ratio	Starting and ending point	Need to calculate the aspect ratio of given contour	Aspect ratio calculated	success

Table 1: Test cases For cursor movements

7.Results

The mouse control program will be operational, and the user can move the cursor, scroll, or click at his will. The amount of change of the position of the cursor along any axis can be changed as per the needs of the user. The mouse control is activated by opening the mouth when the MAR value crosses a certain threshold.

The scroll mode is activated by squinting. The scrolling can be done by moving the head up-down which is called as pitching and by sideways called as yawing. Scroll mode is deactivated by squinting again. The clicking action takes place by winking the eye. Right wink corresponds to right wink and left click corresponds to left wink. The sensitivity of the mouse can be changed accordingly as per the needs of the user. Overall, the project works as required. Though the comfort is not the same as in case of hands-controlled mouse, this project can be used with some ease with some practice.

In this paper, we have overcome many limitations of the traditional eye trackers and their applications like Calibration issues, practical problems while developing an eye tracking system. We also performed two experiments depicting that there does not exist much difference in the accuracy of the Eye Tribe while taking eye gaze inputs of the user wearing or not wearing glasses. The Eye Tribe has a sampling rate of 30 Hz and 60Hz mode and an accuracy of 0.5degree to 1 degree.

The spatial Resolution of the device is 0.1 degree (RMS) and Latency less than 20ms at 60Hz. The operating range lies between 45cm-75cm on an average and provides a tracking area of about 40cmX30cm at 65 cm distance at 30 Hz. The recommended screen size is about 24 inches and outputs Binocular gaze data. Thus, with such high latency as claimed by the Eye Tribe and also proved via the Search Task experiment makes this device highly accurate and advisable to use as eye gaze input. The eye-gaze input system led to a faster pointing time as compared with mouse input, especially for older adults. This result demonstrates that an eye-gaze input system may be able to compensate for the declined motor functions of older adults when using mouse input. Also, during the task where we tested the accuracy of the Eye Tribe, we used two types of data, one with glasses and the other without glasses. So, accuracy of the Eye Tribe in our Lab without the spectacle came out to be 50.3% while without spectacle it was reported to be 47.96%. This indicates an error rate of roughly 2%, which is assumed to be quite good and acceptable.

8.Conclusion and Future Scope

This work can be extended to improve the speed of the system by using better trained models. Also, the system can be made more dynamic by making the change in the position of the cursor, proportional to the amount of rotation of the user's head, i.e., the user can decide, at what rate he wants the position of the cursor to change. Also, future research work can be done on making the ratio more accurate, since the range of the values are the result of the aspect ratios, which is usually small. Hence, to make the algorithm detect the actions more accurately, there can be some modification in the formulae for the aspect ratios used. Also, to make the process of detection of the face easier, some image processing techniques can be used before the model detects the face and features of the face.

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 combines both the mouse functions and keyboard functions, so that users can use our system to achieve almost all of the inputs to the computer without traditional input equipment. 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. According to our TAM questionnaire analysis, the participants considered our eye movement system to be easy to learn. Meanwhile, participants show their interest in using the proposed eye control system to search and browse information. They are looking forward to see more of our research results on the use of eye tracking technique to interact with the computer.

In future, many people who are unable to operate a standard computer mouse or keyboard because of disabilities of their hands or arms, can get possible alternative in multimodal system, which allows controlling a computer without using standard mouse and keyboard. Using head movements to control the cursor across the computer screen and by using the speech for giving the control commands. Automatic speech recognition and head tracking in joint multimodal action are combined to operate the system.

Eye movements provide objective data on how subjects perceive the world and how they react when subjected to different kinds of stimuli, which can be put to use for Researches in Psychology. Eye tracking devices combined with physiological data such as brain imaging can help identify how the information is processed in the brain. Eye tracking can be used to analyze visual development and link it to developmental aspects of neurological functions, neurological diseases and brain damage. Also, reading patterns can be cross referenced with different demographics of people and therefore provide insight into how they gather information.

Human computer interaction allows users to input information in a more natural way into their

computers. Eye tracking can be used as a control medium, like moving the cursor and clicking on icons on the screen, as well as creating adaptive user interfaces, where the computer reacts to the eye gaze of the user and create an interactive environment. Eye tracking is great for coaches who want to train their players on effectively gathering information from the field through simulations. Batsmen's eye movements monitor the moment when the ball is released, make a predictive saccade to the place where they expect it to hit the ground, wait for it to bounce, and follow its trajectory for 100−200ms after the bounce. Learning to analyze an environment quickly can be a valuable skill in air traffic control, radar control, medical X-ray examinations, video surveillance, industrial process control, driving, army or police field work, surgical training and others. Teaching this skill through simulations using an eye tracker can eliminate the time-consuming process of only learning it with experience.

9.References

i. D. H. Yoo, J. H. Kim, B. R. Lee, and M. J. Chung, “Non-contact Eye Gaze Tracking

ii. System by Mapping of Corneal Reflections,” in Fifth IEEE International Conference on Automatic Face and Gesture Recognition (FGR02), 2002, pp. 94-99.

iii. Rafael Barea, Luciano Boquete, Manuel Mazo, and Elena Lpez, “System for assisted mobility using eye movements based on electrooculography,” IEEE TRANSACTIONS ON NEURAL SYSTEMS AND REHABILITATION ENGINEERING, vol. 10, no. 4, pp. 209-217, DECEMBER 2002.

iv. H. Singh and J. Singh, “A Review on Electrooculography,” International Journal of Advanced Engineering Technology, vol. III, no. IV, 2012.

v. K. Irie, B. A. Wilson, and R. D. Jones, “A laser-based eye-tracking system,” Behavior Research Methods, Instruments, & Computers, vol. 34, no. 4, pp. 561-572, 2002.

vi. P Ballard and George C. Stockman, “Computer operation via face orientation,” in Pattern Recognition, 1992. Vol.I. Conference A: Computer Vision and Applications, Proceedings., 11th IAPR International Conference on, 1992, pp. 407-410.

vii. T. Horprasert, Y. Yacoob, and L.S. Davis, “Computing 3-D head orientation from a monocular image sequence,” in Second International Conference on Automatic Face and Gesture Recognition, 1996, pp. 242- 247.

viii. K. Arai and M. Yamaura, “Computer Input with Human Eyes-Only Using Two Purkinje

ix. Images Which Works in a Real-Time Basis without Calibration,” CSC Journals, vol. 1, no. 3, pp. 71-82, 2010.

x. D. Back, “Neural Network Gaze Tracking using Web Camera.,” Linkping University, MS Thesis 2005. [10] R. Gonzalez and R. Woods, Digital Image Processing, 3rd ed.:

xi. Pearson Education, 2009.

SAMPLE CODE

Step 1:

```
import cv2
cam = cv2.VideoCapture(0)
while True:
   _, frame = cam.read()
   cv2.imshow("Eye controlled mouse", frame)
   cv2.waitKey(1)
```

Step2:

```
Import cv2
Import mediapipe as mp
cam=cv2.VideoCapture(0)
face_mesh=mp.solutions.face_mesh.FaceMesh()
while True:
   _, frame = cam.read()
   rgb_frame=cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
   output=face_mesh.process(rgb_frame)
   landmark_points=output.multi_face_landmarks
   print(landmark_points)
   cv2.imshow("Eye controlled mouse", frame)
   cv2.waitKey(1)
```

Step 3:

```
Import cv2
Import mediapipe as mp
cam=cv2.VideoCapture(0)
face_mesh=mp.solutions.face_mesh.FaceMesh( refine_landmark=True)
while True:
   _, frame = cam.read()
   frame=cv2.flip(frame,1)
   rgb_frame=cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
   output=face_mesh.process(rgb_frame)
```

```
    landmark_points=output.multi_face_landmarks
print(landmark_points)
if landmark_points:
    landmarks=landmark_points[0].landmark
    frame_h,frame_w,_=frame.shape
    for landmark in landmarks:
        x=int(landmark.x*frame_w)
        y=int(landmark.y*frame_h)
        cv2.circle(frame, (x, y), 3, (0,255,0)).
    cv2.imshow("Eye controlled mouse", frame)
    cv2.waitKey(1)
```

Step4:

```
Import cv2
Import mediapipe as mp
import pyautogui
cam=cv2.VideoCapture(0)
face_mesh=mp.solutions.face_mesh.FaceMesh( refine_landmark=True)
screen_w,screen_h=pyautogui,size()
while True:
    _, frame = cam.read()
    frame=cv2.flip(frame,1)
    rgb_frame=cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    output=face_mesh.process(rgb_frame)
    landmark_points=output.multi_face_landmarks
if landmark_points:
    landmarks=landmark_points[0].landmark
    frame_h,frame_w,_=frame.shape
    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=screen_w/frame_w*x

screen_y=screen_h/frame_h*y

pyautogui.moveTo(screen_x,screen_y)

cv2.imshow("Eye controlled mouse", frame)

cv2.waitKey(1)

Step5:

Import cv2

Import mediapipe as mp

import pyautogui

cam=cv2.VideoCapture(0)

face_mesh=mp.solutions.face_mesh.FaceMesh(refine_landmark=True)

screen_w,screen_h=pyautogui,size()

while True:

_, frame = cam.read()

frame=cv2.flip(frame,1)

rgb_frame=cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

output=face_mesh.process(rgb_frame)

landmark_points=output.multi_face_landmarks

if landmark_points:

landmarks=landmark_points[0].landmark

frame_h,frame_w,_=frame.shape

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=screen_w/frame_w*x

screen_y=screen_h/frame_h*y

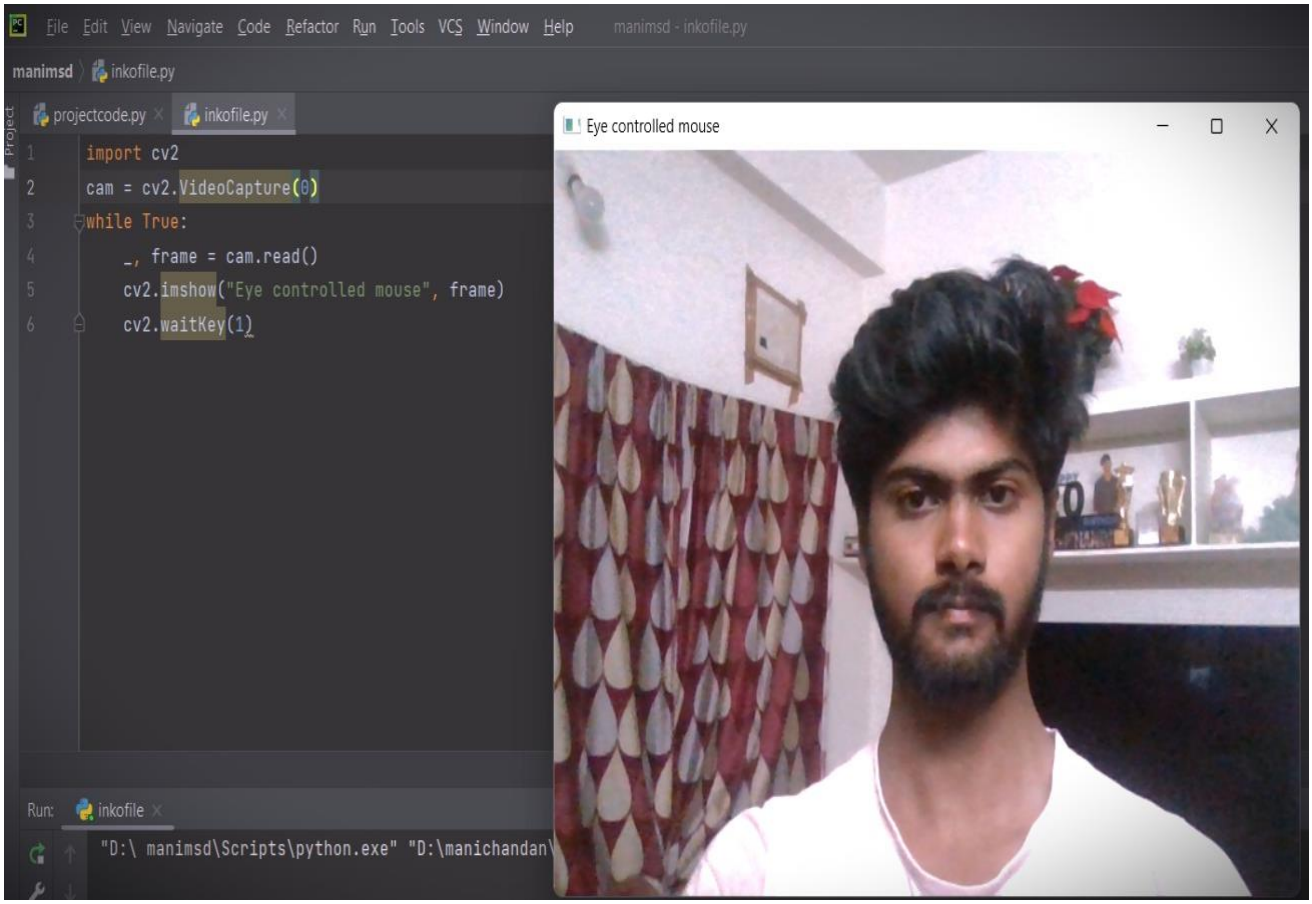
pyautogui.moveTo(screen_x,screen_y)

left = [landmarks[145], landmarks[159]]

```
    for landmark in left:
        x = int(landmark.x * frame_w)
        y = int(landmark.y * frame_h)
        cv2.circle(frame, (x, y), 3, (0, 255, 255))
    if (left[0].y - left[1].y) < 0.004:
        pyautogui.click()
        pyautogui.sleep(1)
cv2.imshow("Eye controlled mouse", frame)
cv2.waitkey(1)
```

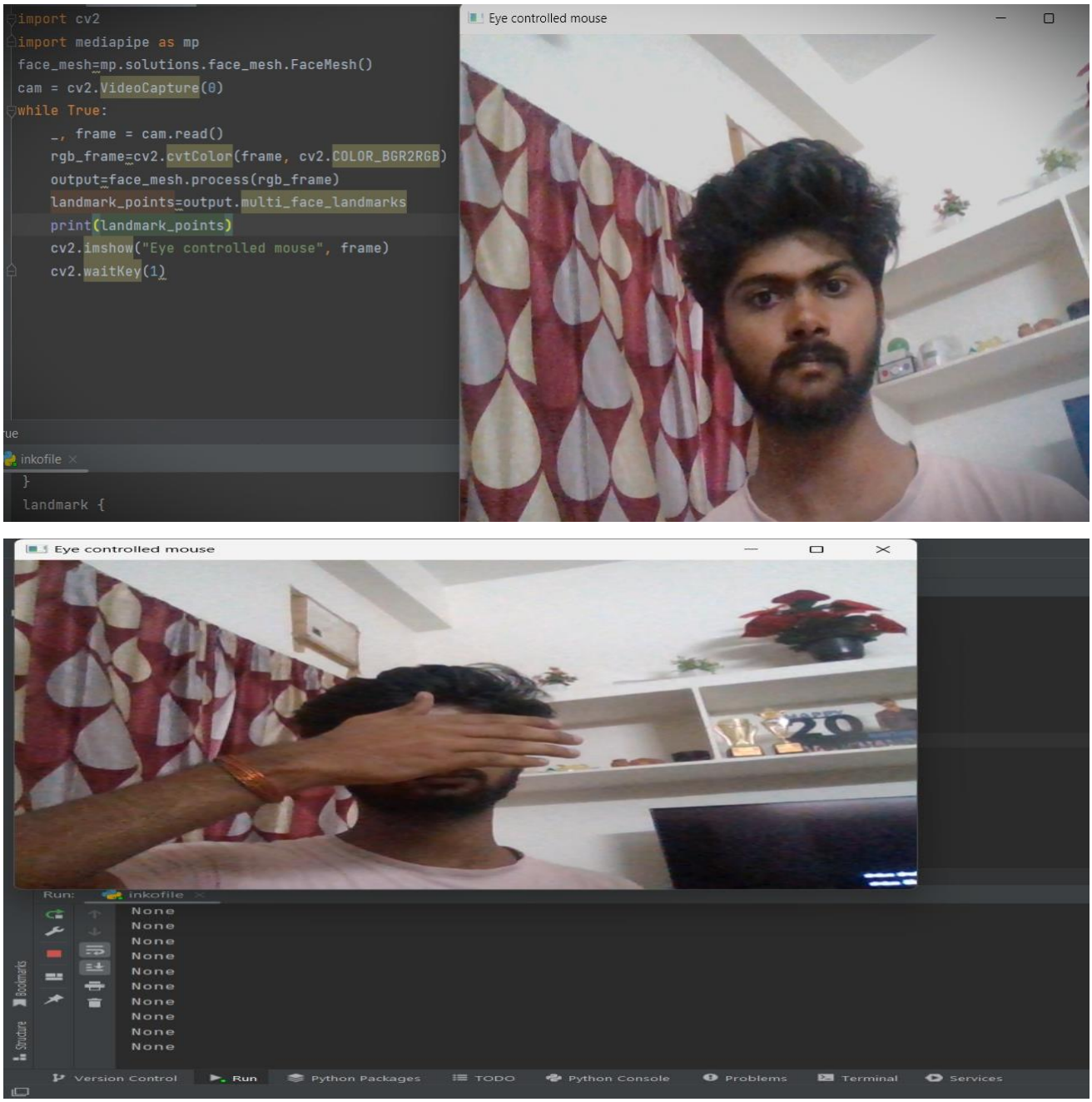
OUTPUT SCREENS

STEP 1: To open the cam and see ourself



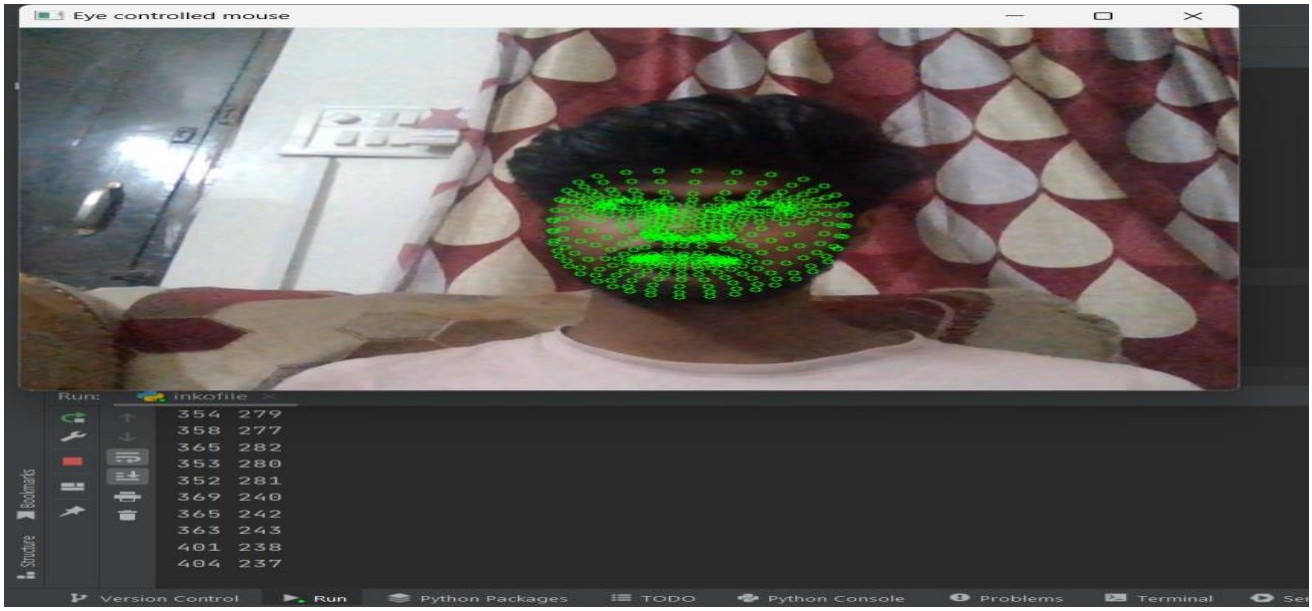
- Importing the OpenCV to open the webcam.
- Read the camera and telling the cv2 to capture the video.
- Creating a while loop to run for a while.
- Inside while we call the camera to read the frames.
- Again, calling the cv2 to show image.

STEP 2: To detect the face

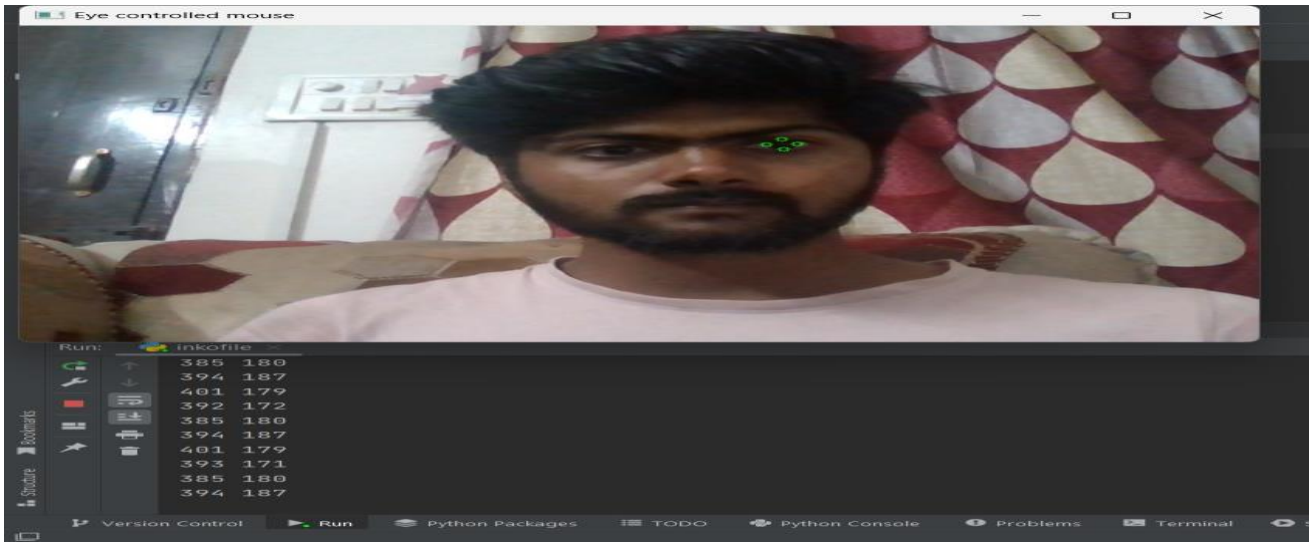


- Import mediapipe to detect the face eventually eye.
- Declare facemash.
- Detect face in colors.
- Create and output for facemesh.
- Declare landmarks point (which you get from output).

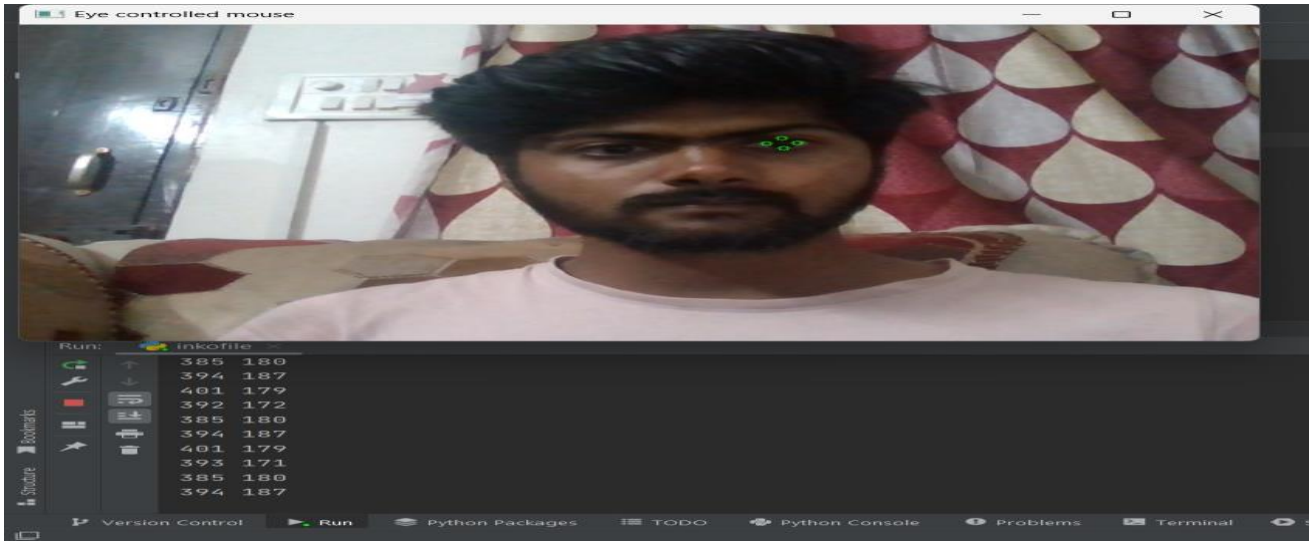
STEP 3: To show face and Landmarks



- If there are landmark points, we need to find them (here we are taking 0 to identify only one face).
- To get landmarks we need to get coordinates (because to draw something).
- We care about x and y as we are working on 2-D.
- Need to know the width and height of the frame.
- To draw a circle we need integer numbers, center, radius, colors.
- Landmarks gives 478 landmark points.
- To detect only iris, we give a range.

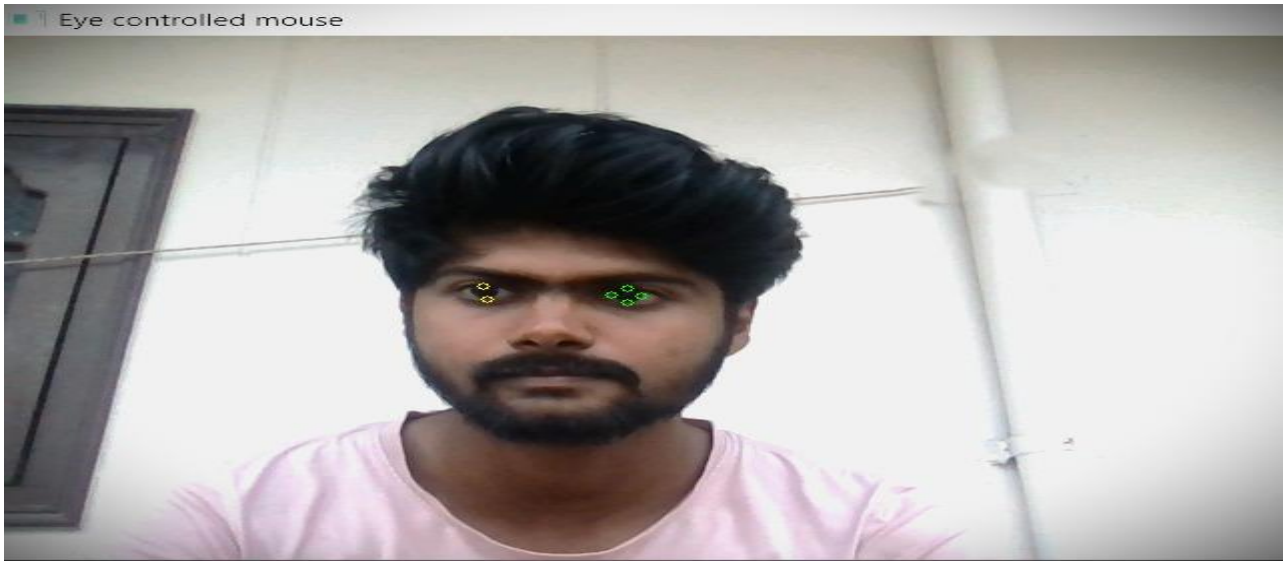


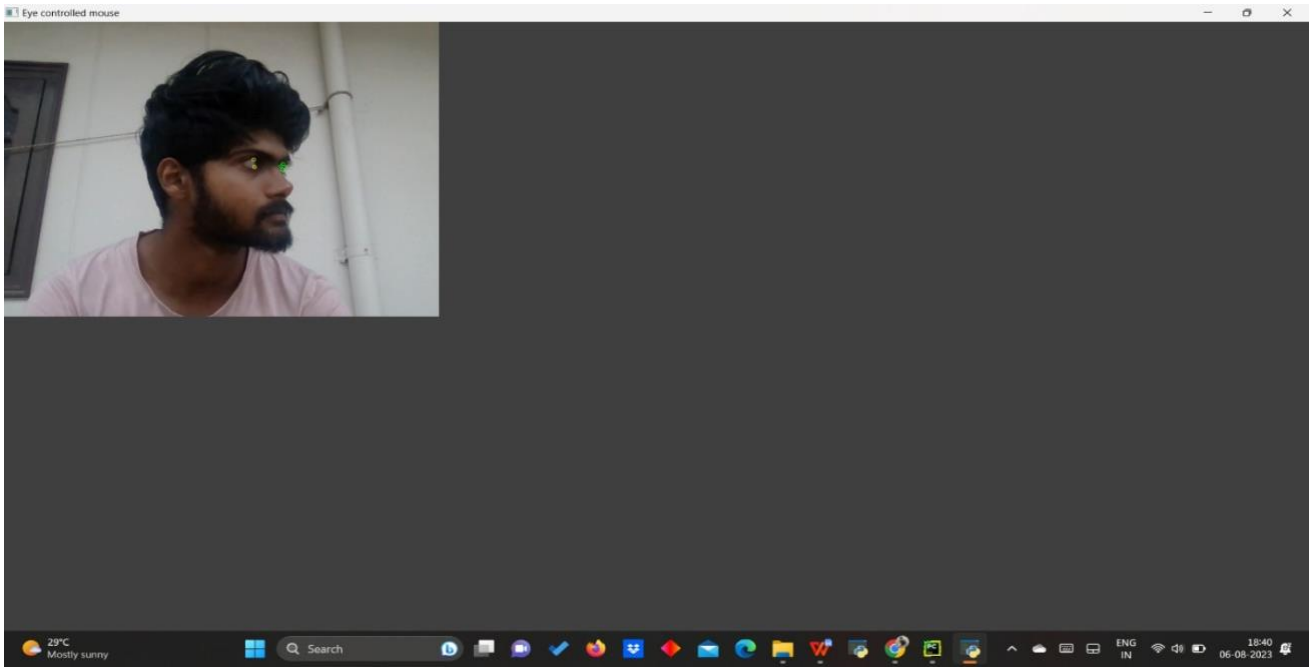
STEP 4: To move the Cursor



- Import pyautogui.
- Enumerate gives you id of index and landmark point.
- Selecting id, to move the cursor.
- CTRL+F2 to close the frame.
- Get the screen size.

STEP 5: To click





- To click is to wink the eye.
- To blink the eye, we need to get the bottom and top eye lid (use landmarks).
- When the two landmarks touch the eye blinks.
- Yellow colored is to click and green colored to move the cursor.