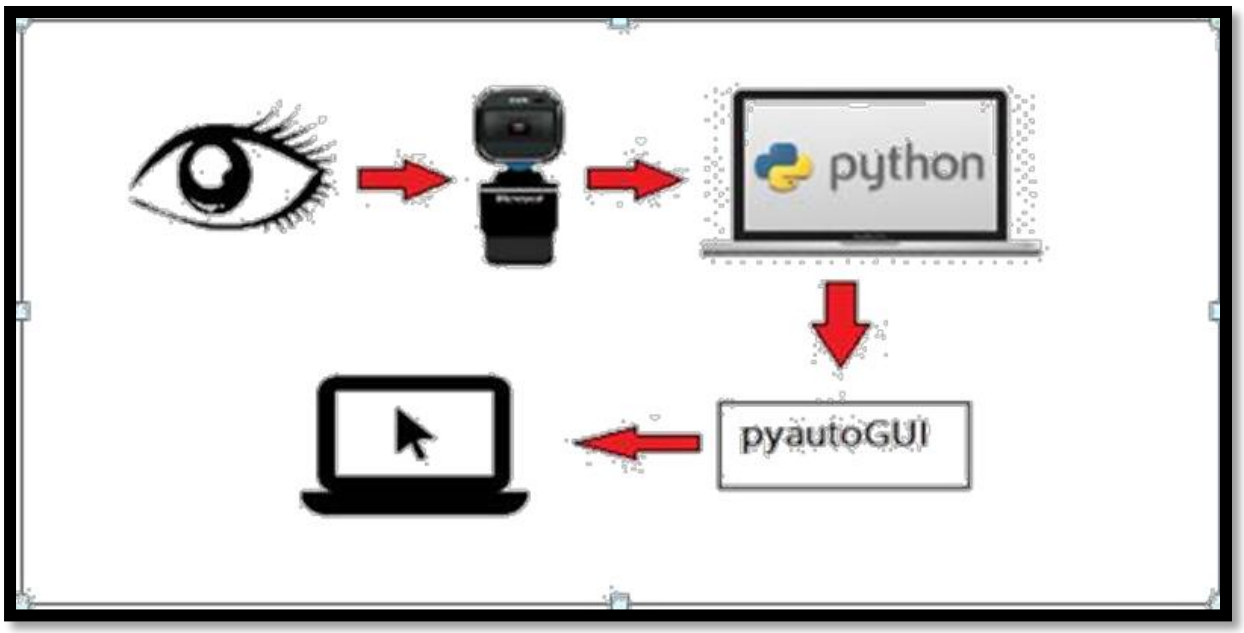
❖ INTRODUCTION

- Fundamentally paralyzing diseases like paraplegia, which renders a person unable to move from the neck down, are becoming more common in today's society.
- In the majority of OECD (Organization for Economic Cooperation and Development) nations, women are more likely than men to experience disability.
- Their eyes are the only organ that can produce various actions. 518 million persons out of a population of 7 billion reported having a disability in the 2011 Census. Around 10% (or 650 million) of the world's population, as of February 7, 2018, has a disability.
- Many people with Amyotrophic lateral sclerosis (ALS) [3] or those who are paralysed are unable to perform routine daily tasks on computers. Even when it comes to eating, they require assistance from someone else to feed them.
- For their daily tasks, these people require assistance. At the moment, people with impairments frequently type on keyboards by holding long sticks in their mouths.
- The method we offer will enable people with disabilities to lead independent lives. They will have the opportunity to amuse themselves, mingle with others, and live their lives.
- The development of innovative and cutting-edge HCI techniques is accelerating. This study area is actively being worked on by many professionals.
- Human eyes contain a wealth of information that can be collected and applied in a variety of ways . (i.e. interacting with Computers).
- Eye movement exhibits an a person's area of interest. The goal of tracking eye motions is to monitor human eye movements.
- By recording eye movements and using them as control signals, direct interaction with interfaces can be made possible without the need for a keyboard or mouse.

❖ METHODOLOGY

- It relates to the field of Human-Computer Interaction (HCI) and demonstrates how a low-cost eye tracking solution can be created for patients with disabilities by enhancing current open source frameworks used for Computer Vision and HCI. The system model and overview are shown in Fig.



- The System prototype uses camera input to identify and track the user's pupil in real-time . Computers or
- microcontrollers can use this "tracking" information to perform a variety of tasks.
- One of these tasks is to track the pupil-movement and then store that tracked eye movement to control a computer's mouse pointer, allowing someone with a disability like, say, Amyotrophic Lateral Sclerosis, to use it to communicate with others.
- It comes with a high resolution web camera that is strategically placed, as well as an open-platform software module that is simple to install and is compatible with all current laptops and desktop computers.
- This system can be viewed as a seamless movement between the concept, design, and proof of concept phases.
- It involves implementing portions of research papers and collaborating with the open-source community to design and create a prototype, all while making sure that only open-source, affordable, easily accessible, and commercially off the shelf (COTS) items are used.

❖ MODELING

Python was used to design the mouse system, and the following Python modules were imported to make the system operate.

- A Python extension module called NumPy It offers quick and effective actions on collections of related data.
- Scipy is a Python library that is open-source and used for technical and scientific computing.
- OpenCV is a collection of programming tools with a real-time computer vision major focus.
- PyautoGUI is a Python-based, cross-platform GUI automation module. This allows you to automate
- computer chores by controlling the mouse and keyboard as well as performing simple picture recognition.

A. Use Case Diagrams

Use case diagram for 0 indicates that the system completes the subsequent stages.

1) Software is active.

1) Turn on the laptop's webcam and display a picture of a person.

2) Face detection is carried out.

3) The system recognises a person by their eyes.

4) After performing the aforementioned action, perform the subsequent procedure.

5) The system then uses a laptop's webcam to recognize eyes and a face.

6) Students discovered that a person can now control the mouse cursor with eye movements. The computer's home screen displays the progress of the core.

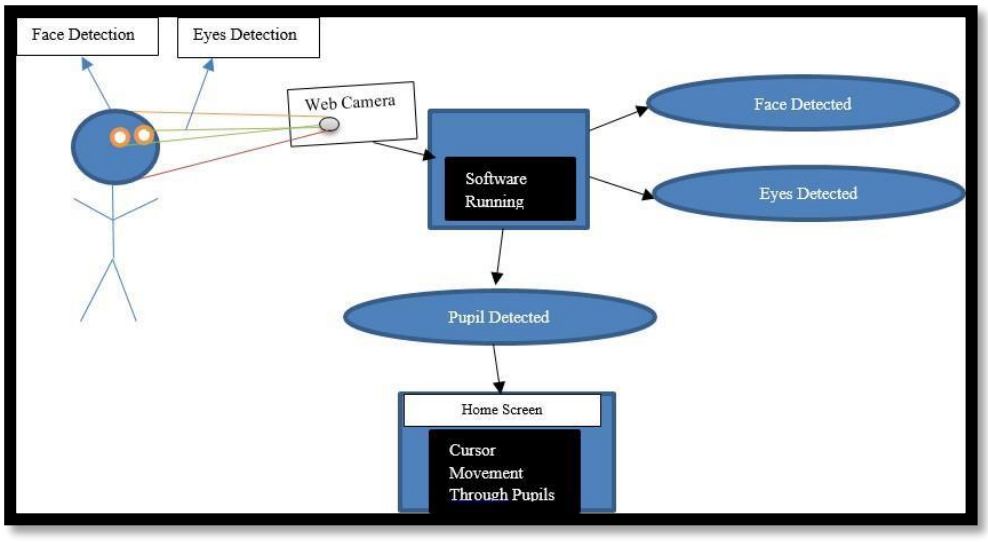
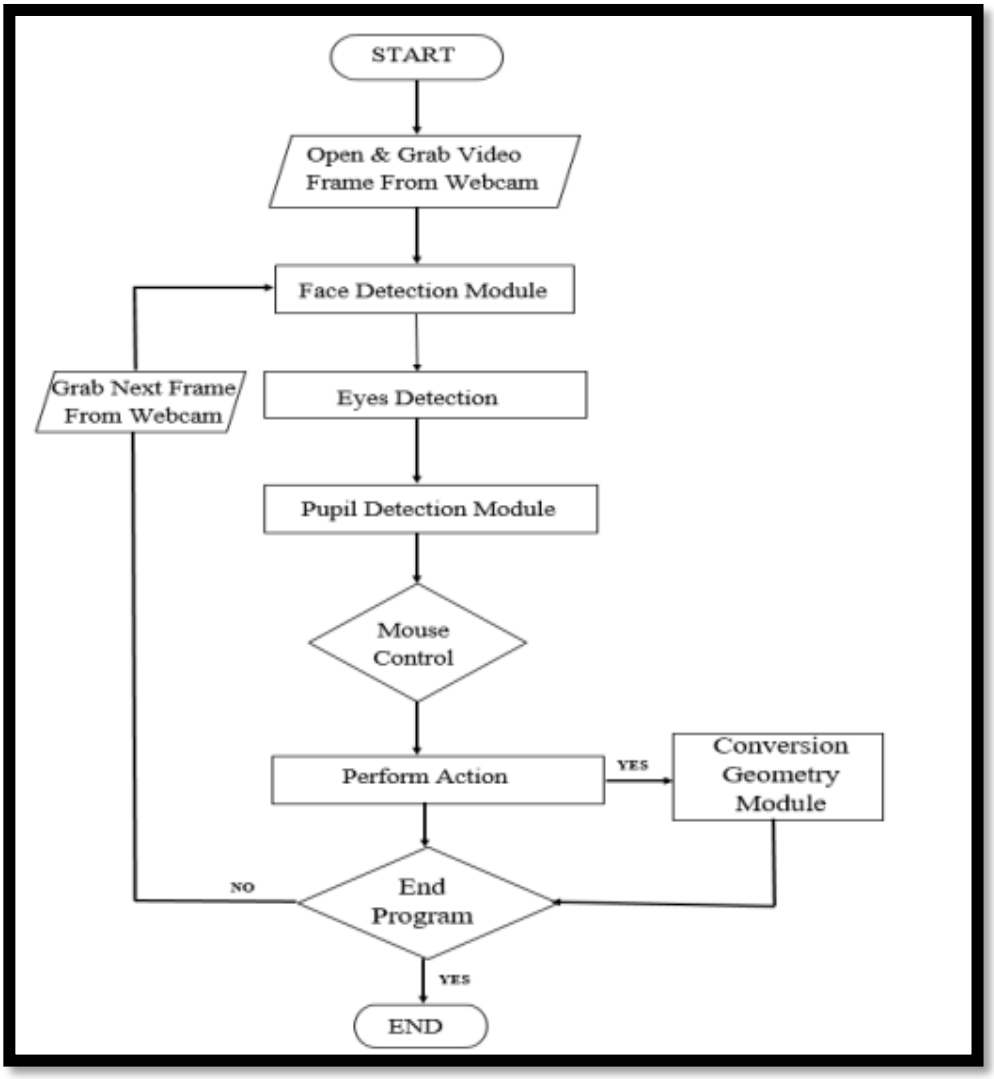


Figure: System Use Case Diagram



B. Activity Diagram

The order in which activities are held in the system is depicted in Fig.

Figure epicts the system's operation. The I Mouse software has the following phases from beginning to end.

1) Switch on the webcam and record some footage.

2) The system takes a step and finds the face.

3) The system detects the presence of eyes.

4) The mechanism locates the pupil of the eye.

5) The system will find the eyes and carry out geometric translations using only the image of the user's face from the webcam.

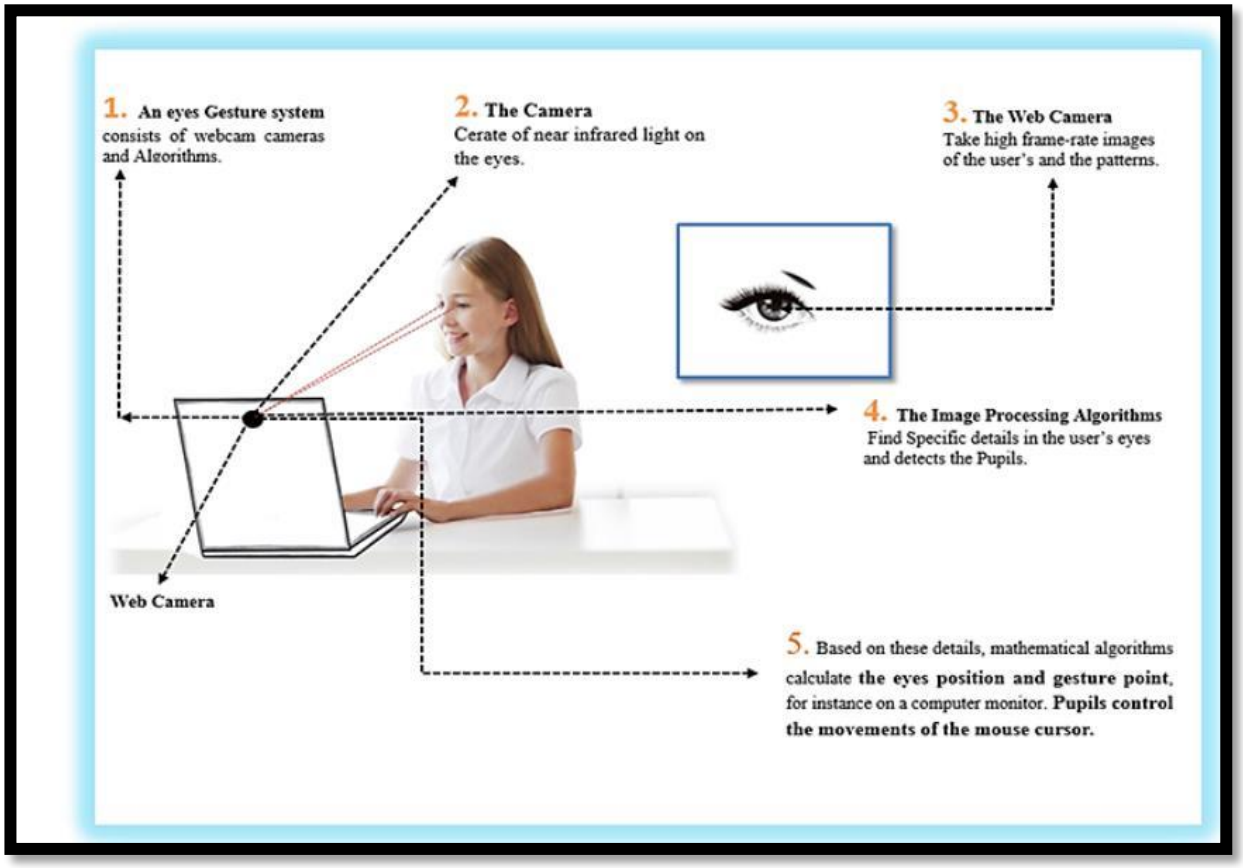


Figure: System Functionality Diagram

The user's interactions with the system are depicted in Fig. in order. The six fundamental modules of our system are elaborated in Fig.'s system sequence diagram.

Using detection techniques, the system uses the webcam to identify a person's pupil in the first module. The machine then finds the face. The mechanism then finds and seizes the eyes.

The machine then finds the pupils. The system begins moving the mouse cursor in the last module by monitoring pupil movement.

❖ IMPLEMENTATION

Using Python, an image is created. First, it opens the camera and starts taking video. It then selects a frame and changes it to a grayscale image because it converts images to binary form, making it easy to find things. The face is detected using Haar-cascade.

Haar-cascade detects items from other photos after training on many positive and negative images. It will crop the frame and pass it on for additional processing after detecting the face. The Haar-cascade algorithm will detect eyeballs and trim the frame. Eye-cascade The Haar cascade recognises eyes.

A numpy-supported four-variable array gives us x, y, w, and h all at once. The camera detects eyeballs from x and y; w is the width, and h is the height. These variables construct a rectangle around the eye and crop the image The rectangle begins at x and finishes at x+w horizontally and y and y+h vertically.

For further processing, the application will blur the little image. It then finds Then use the Hough Transform to draw a circle around the pupil. Sometimes the camera sees black rings around the eyes as pupils.

The technology detects circular dark patches in the rectangle's centre to fix this. The camera will only track the eye's pupil this way. Define x and y using pyauto gui for cursor movement. Y moves the mouse vertically, x horizontally. Both of them start with a random value, so when the code runs, the mouse starts moving. Check the gap between two frames to detect eye movement. Human eyes move somewhat.

To fix this, the eye is considered motionless if its position difference is less than 5 pixels. If the eye's horizontal location is larger than 5 pixels and its vertical position is less than 5 pixels, it's travelling horizontally.

The eye moves diagonally if its horizontal and vertical positions differ by more than 5 pixels. The mouse cursor moves vertically, horizontally, and diagonally when we move our eyes

B. System Analysis & Evaluation :

The Haar-cascade and Hough Transform contour detecting algorithm's findings are presented first. B. The System Analysis & Evaluation: We begin by showcasing the outcomes of the operational contour detection technique, which makes use of the Haar-cascade functions and Hough Transform.

C. Harr-cascade Algorithm:

The system performs two tasks in this algorithm. determine the person's face and eyes The face-cascade function

Face-cascade is used to initially identify the user's face in an image. It crops the image and run scan around the face for later processing. The user's eyes are picked up from the image once the face has been extracted. as depicted.

Because both eyes move simultaneously, tracking can be done by looking at just one eye's motions. This programme selects the user's left eye (the right eye when the image is flipped in a camera) and tracks eye movement. It crops this picture by drawing a box around it.

D. Hough Transform Algorithm :

From this cropped image, a circle is drawn around the Hough Transform pupil to track its movements. This system identifies the pupil, ignores other dark spots, and tracks only the pupil of the eye, as shown in Fig. 10 below.

After that, each frame's eye coordinates are located. The eye is regarded as moving if its coordinates change, otherwise they are regarded as remaining static. This entire processing process takes less than one second to complete. This means that this project can be applied to real-world circumstances.

E. Mouse vs. Eye Histograms:

Please note that the readings were collected under optimum lighting settings and may differ in other circumstances if pupil detection is inaccurate in the histograms for mouse vs. eye control, which are displayed below. The 0and 0 provides an illustration of this.

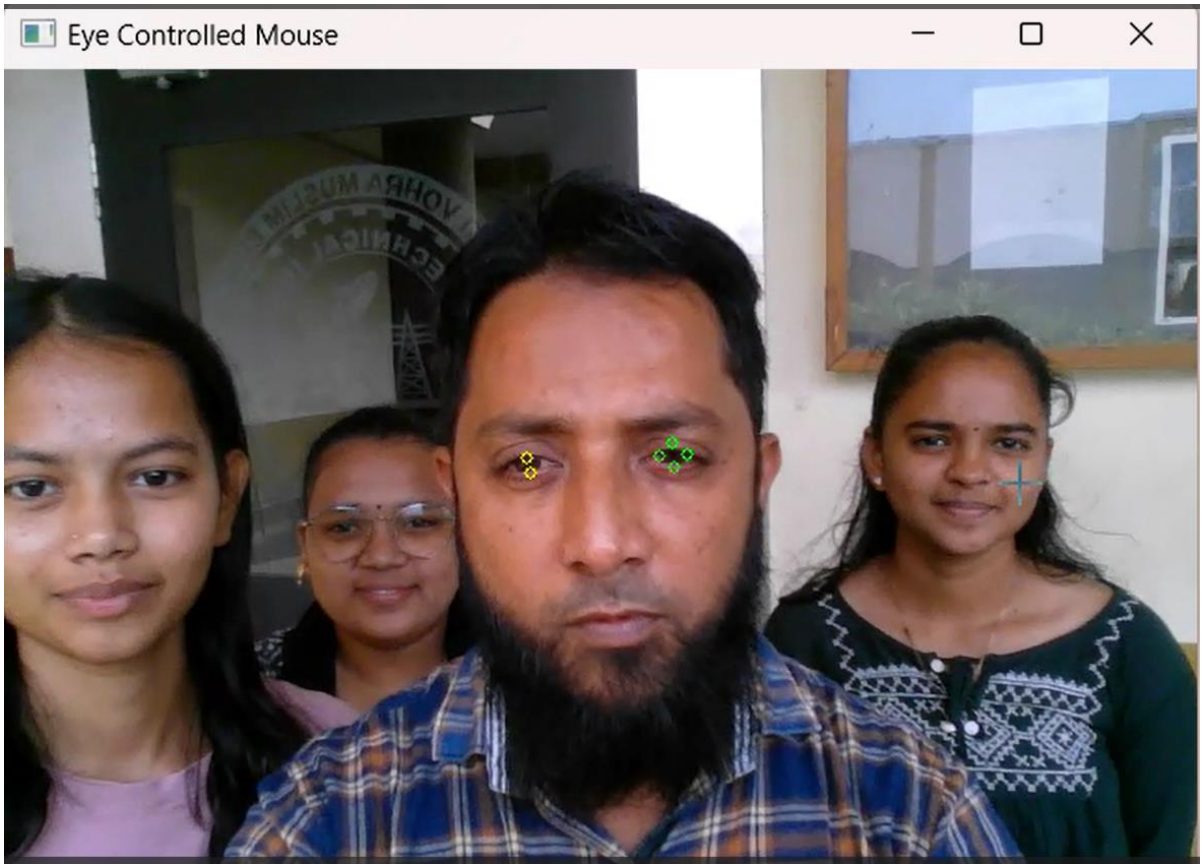
Please note that the results for 0 and 0 are rounded to the closest millisecond. We can see that after mastery is attained, the user can carry out the identical tasks in roughly the same amount of time as they can be carried out with a mouse.

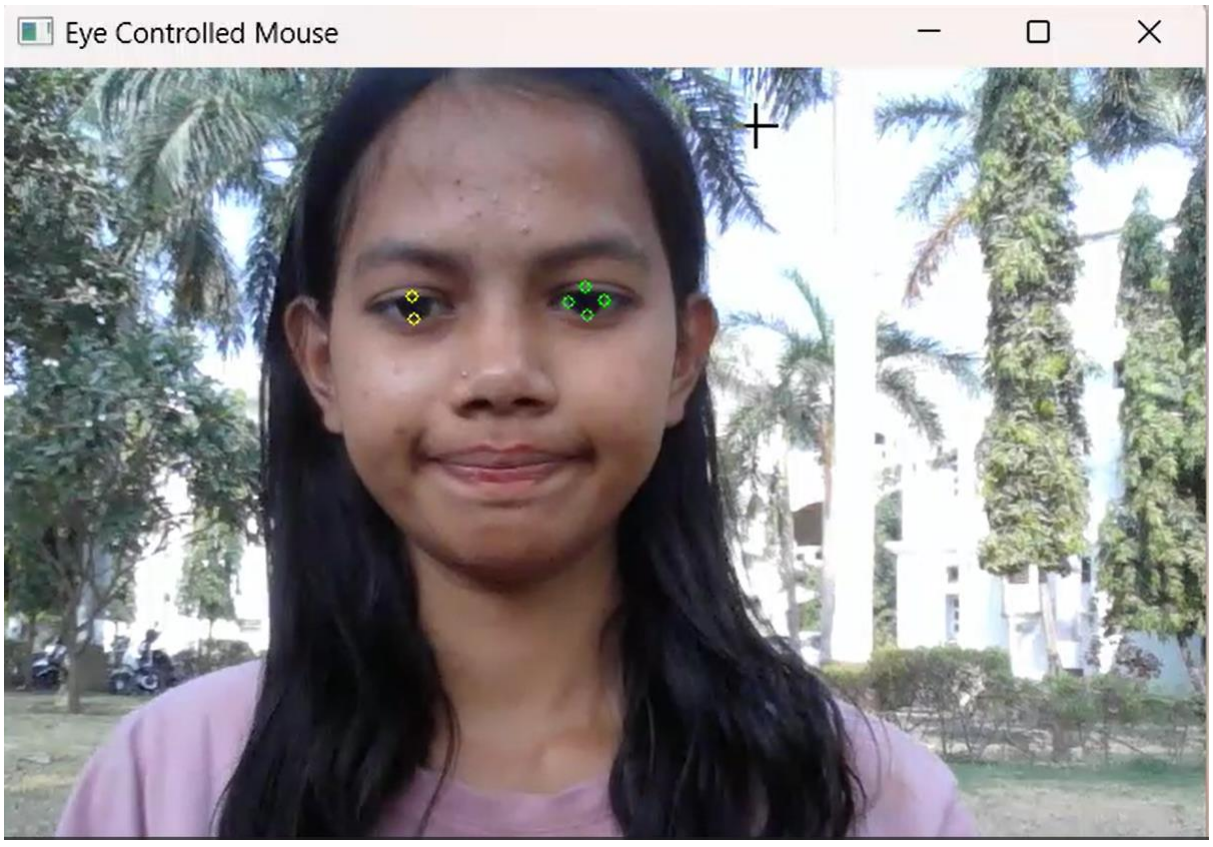
After tracking the pupil, the coordinates are saved in variables, and the mouse cursor is moved in accordance with changes in these values and the detection of blinks. At this point, the eye mouse begins functioning properly.

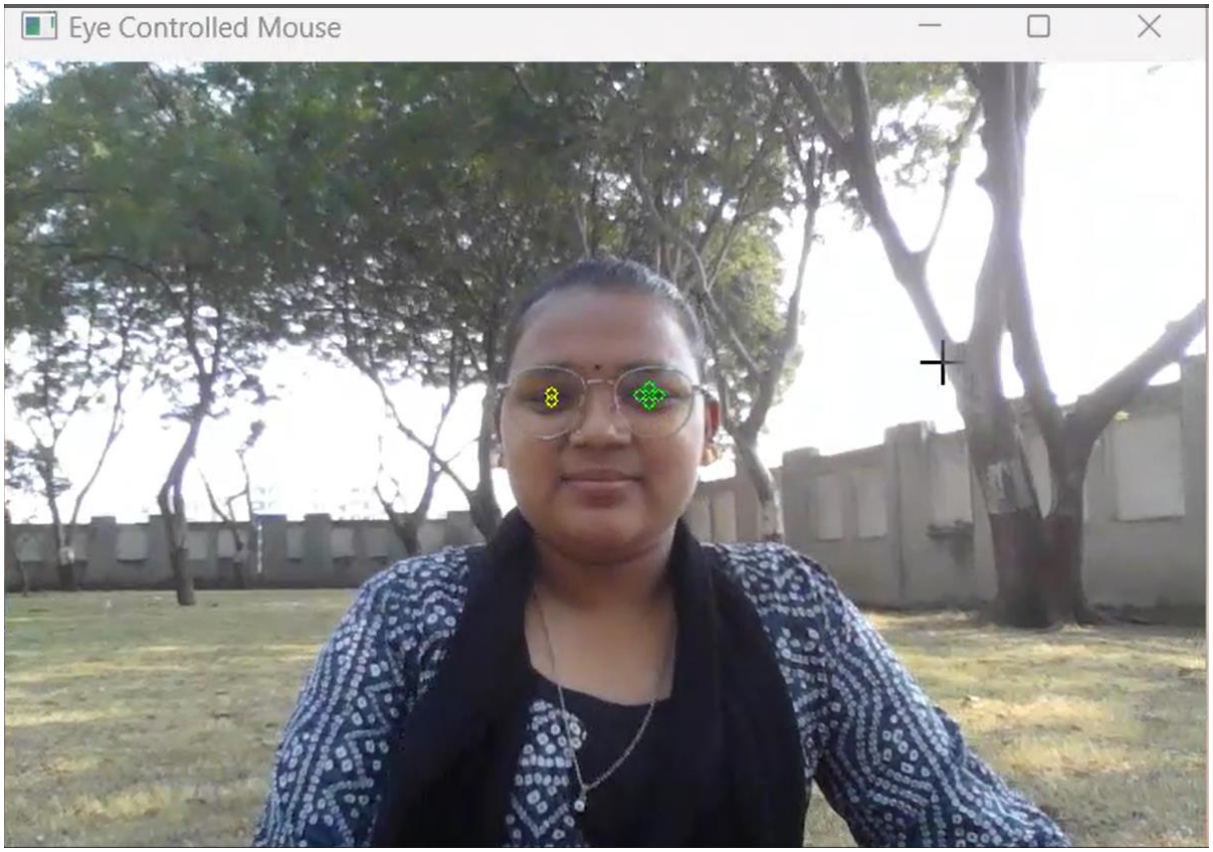
❖ CODE:

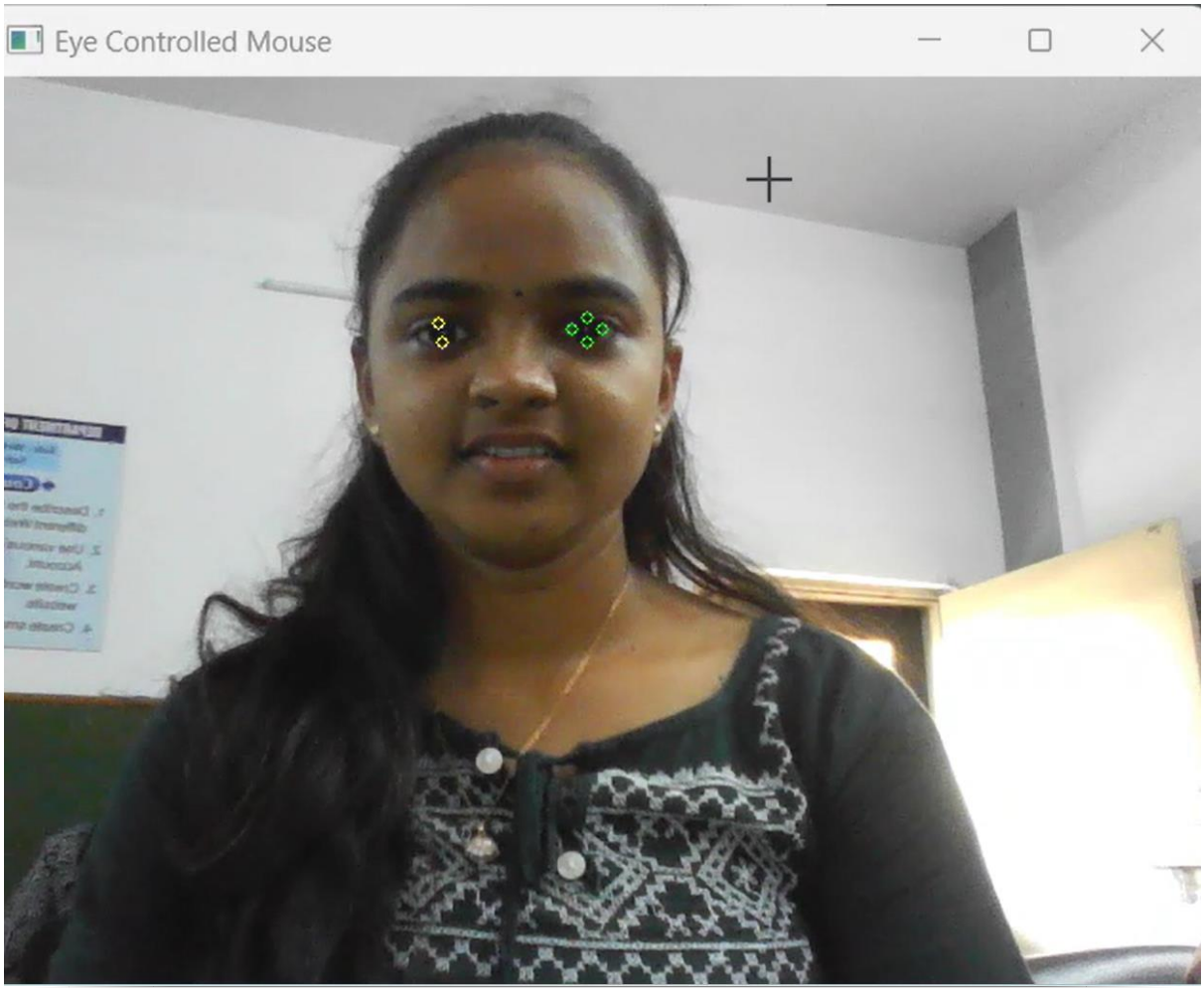
```
import cv2
import mediapipe as mp
import pyautogui
cam = cv2.VideoCapture(0)
face_mesh = mp.solutions.face_mesh.FaceMesh(refine_landmarks=True)
screen_w, screen_h = pyautogui.size()
while True:
    _, frame = cam.read()
    frame = cv2.flip(frame, 1)
    rgb_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    output = face_mesh.process(rgb_frame)
    landmark_points = output.multi_face_landmarks
    frame_h, frame_w, _ = frame.shape
    if landmark_points:
        landmarks = landmark_points[0].landmark
        for id, landmark in enumerate(landmarks[474:478]):
            x = int(landmark.x * frame_w)
            y = int(landmark.y * frame_h)
            cv2.circle(frame, (x, y), 3, (0, 255, 0))
            if id == 1:
                screen_x = screen_w * landmark.x
                screen_y = screen_h * landmark.y
                pyautogui.moveTo(screen_x, screen_y)
        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:**






Eye Controlled Mouse


Eye Controlled Mouse

❖ CONCLUSION

This device seeks to provide a low-cost eye-tracker that will enable the user to control a computer system's mouse cursor. The system is simple to use and cost-effective, relying just on a laptop camera and C++ and Python programming language software modules.

To adjust the interface or just extract spatial attention data for the objectives specified in the "future applications" section, the spatial field of view history can also be rendered on the world process as needed, indicating eye movements and where the user spent a lot of time gazing.

Finally, we point out that the project can be used in a variety of environmental settings with just minor adjustments to the brightness and contrast needed to retain its durability. This is a remarkable accomplishment for such a cheap eye-tracking technology

❖ REFERENCES

- Disabled persons Ratio [online] Available At: https://www.disabled-world.com/disability/statistics/ (Accessed 2 Jan 2018)
- The Eye Writer Project [online] Available At: http://www.eyewriter.org(2009-Present)
- Muhammad Usman Ghani, Sarah Chaudhry "Gaze Pointer: A Real Time Mouse Pointer Control Implementation Based On Eye GazeTracking"/ http://myweb.sabanciuniv.edu/ghani/files/2015/02/GazePoint er_INMIC2013.pdf [6 February 2014]
- Schmidt, Jochsen, Vogt and Nieman "Calibration–free hand–eye calibration: a structure–from–motion approach." Pattern Recognition. Springer Berlin Heidelberg, 2005. 67-74.) (2005)
- Prajakta Tangade, Shital Musale "A Review Paper on Mouse Pointer Movement Using Eye Tracking System and Voice Recognition" http://www.ijeert.org/pdf/v2-i8/20.pdf (08/11/2014)
- PUPIL: constructing the space of visual attention. Diss. Massachusetts Institute of Technology, 2012.) Kassner, Phillip, and Patera (2012)
- IEEE paper on "Differences in the infrared bright pupil response of human eyes" by Karlene Nguyen, Cindy Wagner, David Koons, Myron Flickner (2009)
- "An eye tracking algorithm based on Hough transform"https://ieeexplore.ieee.org/document/8408915/ (12.5.2018
- Eye gaze location detection based on iris tracking with web camera: https://ieeexplore.ieee.org/document/8404214/ (5.7.2018)
- "An image-based eye controlled assistive system for paralytic patients" https://ieeexplore.ieee.org/document/8066549/ (2017)