VISION MOUSE FOR HANDICAP

A PROJECT REPORT

Submitted

by

NAME OF THE CANDIDATES

NARENDIRAN A

SANKARA NARAYANAN K

SAILESH R

in partial fulfilment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

RAJALAKSHMI ENGINEERING COLLEGE

RAJALAKSHMI NAGAR

THANDALAM

CHENNAI 602 105

APRIL 2024

RAJALAKSHMI ENGINEERING COLLEGE

CHENNAI – 602105

BONAFIDE CERTIFICATE

Certified that this project report **"VISION MOUSE FOR HANDICAP"** is the bonafide work of **"NARENDIRAN A (211001066), SANKARA NARAYANAN K (211001092) AND SAILESH R (211001086)"** who carried out the project work under my supervision.

Dr. Priya Vijay
HEAD OF THE DEPARTMENT
Professor and Head
Department of
Information Technology
Rajalakshmi Engineering College
Rajalakshmi Nagar
Thandalam
Chennai - 602105

Mrs.P. Mahalakshmi
SUPERVISOR
Assistant Professor
Department of
Information Technology
Rajalakshmi Engineering College
Rajalakshmi Nagar
Thandalam
Chennai - 602105

ABSTRACT

Advancements in technology continue to revolutionize the way we interact with our devices, and one such innovation that holds immense potential for transforming accessibility is the development of an eye-controlled mouse. This groundbreaking project introduces a novel method of computer interaction, enabling individuals to navigate and operate their computers using only their eyes. By leveraging the capabilities of a camera to monitor facial movements, this system detects specific eye gestures such as blinks or controlled eye closures, translating them into mouse movements and clicks. What emerges is a seamless and intuitive interface that empowers users to interact with their computers hands-free, transcending the limitations imposed by traditional input devices . At its core, this eye-controlled mouse is a testament to the power of inclusive design. By catering to the needs of individuals with disabilities, it promotes inclusivity in technology, ensuring that everyone, regardless of physical limitations, can access and utilize digital resources with ease. For those who face challenges with conventional input methods due to motor impairments or other disabilities, this innovation represents a gateway to a world of possibilities. It provides a means of independent computer usage, granting users autonomy over their digital experiences and fostering a sense of empowerment . One of the key advantages of this eye-controlled mouse is its customizable sensitivity, allowing users to fine-tune the system according to their individual preferences and capabilities. This level of personalization ensures a tailored user experience, maximizing comfort and usability for each individual. Whether adjusting the sensitivity to accommodate subtle eye movements or refining the response time to suit specific needs, users have the flexibility to optimize their interaction with the technology.

Moreover, the implications of this innovation extend far beyond mere convenience. By enabling hands-free computer interaction, it facilitates access to essential tools and resources for communication, education, and work. Individuals with disabilities are empowered to engage with digital content, communicate with others, and pursue educational and professional endeavors with greater independence and efficiency. This not only enhances their quality of life but also fosters greater societal inclusion by breaking down barriers to participation in the digital age . Furthermore, the implementation of an eye-controlled mouse has broader societal implications, particularly in terms of reducing reliance on caretakers and promoting self-reliance among individuals with disabilities. By equipping users with the tools they need to navigate their digital environments independently, this innovation fosters greater autonomy and self-sufficiency. This, in turn, contributes to a more equitable and inclusive society where everyone has the opportunity to harness the benefits of technology to their fullest extent.

In conclusion, the development of an eye-controlled mouse represents a significant step forward in the quest for inclusive technology. By enabling hands-free computer interaction through eye movements, this innovation empowers individuals with disabilities to access and engage with digital resources in ways previously unimaginable. Through customizable sensitivity and personalized user experiences, it ensures a seamless and intuitive interface tailored to the needs of each individual. Ultimately, by breaking down barriers to access and promoting greater independence, this technology paves the way for a more equitable and inclusive society where everyone can participate fully in the digital age.

TABLE OF CONTENTS

LIST OF FIGURES

CHATPER 1
INTROUCTION

Introduction To Vision Mouse For Handicap

In today's interconnected world, technology serves as a gateway to communication, education, employment, and social interaction. Yet, for individuals grappling with physical disabilities, the digital landscape often presents formidable barriers. Conventional input devices like keyboards and mice, while ubiquitous, can pose insurmountable challenges for those with limited mobility or dexterity. Recognizing this disparity, our project endeavors to bridge the gap through the development of the Vision Mouse for Handicap – an innovative solution poised to revolutionize computer interaction for individuals with disabilities. Our primary objective is clear: to empower individuals with physical disabilities to navigate their computers with unparalleled ease and precision using only their eyes. Leveraging state-of-the-art eye-tracking technology, our vision is to create a seamless interface that allows users to control the mouse cursor on their screen with unprecedented accuracy. By eliminating the need for physical input devices, the Vision Mouse for Handicap not only enhances accessibility but also fosters independence and autonomy for users, thereby enriching their overall quality of life.

The motivation driving our project is deeply rooted in the principles of inclusivity and equity. In an era marked by rapid technological advancement, it is imperative that we strive to ensure that no individual is left behind. By developing a hands-free solution tailored to the unique needs of individuals with physical disabilities, we seek to dismantle barriers and empower users to fully participate in the digital realm. Our commitment to accessibility extends beyond mere functionality; it encompasses a broader vision of social inclusion and empowerment for all. The scope of the Vision Mouse for Handicap project is comprehensive yet focused. Our development efforts center on creating an intuitive and user-friendly application that seamlessly integrates eye-tracking technology into the computing experience. Through the implementation of sophisticated algorithms and machine learning techniques, we aim to accurately track users' eye movements and translate them into precise mouse movements and clicks. Furthermore, the application will feature customizable settings to accommodate individual preferences and optimize user comfort and efficiency.

In summary, the Vision Mouse for Handicap project represents a paradigm shift in the field of assistive technology. By harnessing the transformative potential of eye-tracking technology, we aspire to empower individuals with physical disabilities to navigate their digital environments with unparalleled freedom and confidence. Through our unwavering dedication to accessibility and inclusivity, we believe that the Vision Mouse for Handicap has the capacity to not only enhance individual lives but also pave the way for a more equitable and inclusive society.

Open CV :

Opencv is a huge open-source library for computer vision, machine learning, and image processing. Now, it plays a major role in real-time operation which is very important in today's systems. By using it, one can process images and videos to identify objects, faces, or even the handwriting of a human.

When it is integrated with various libraries, such as NumPy, python is capable of processing the opencv array structure for analysis. To Identify an image pattern and its various features we use vector space and perform mathematical operations on these features.

The first OpenCV version was 1.0. OpenCV is released under a BSD license and hence it's free for both academic and commercial use. It has C++, C, Python, and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. When opencv was designed the main focus was real-time applications for computational efficiency. All things are written in optimized C/C++ to take advantage of multi-core processing.

MediaPipe Library :

MediaPipe is an open-source framework for building pipelines to perform computer vision inference over arbitrary sensory data such as video or audio. Using MediaPipe, such a perception pipeline can be built as a graph of modular components. MediaPipe is currently in alpha at v0.7, and there may still be breaking API changes. Stable APIs are expected by v1.0.

The MediaPipe framework is mainly used for rapid prototyping of perception pipelines with AI models for inferencing and other reusable components. It also facilitates the deployment of computer vision applications into demos and applications on different hardware platforms. The configuration language and evaluation tools enable teams to incrementally improve computer vision pipeline.

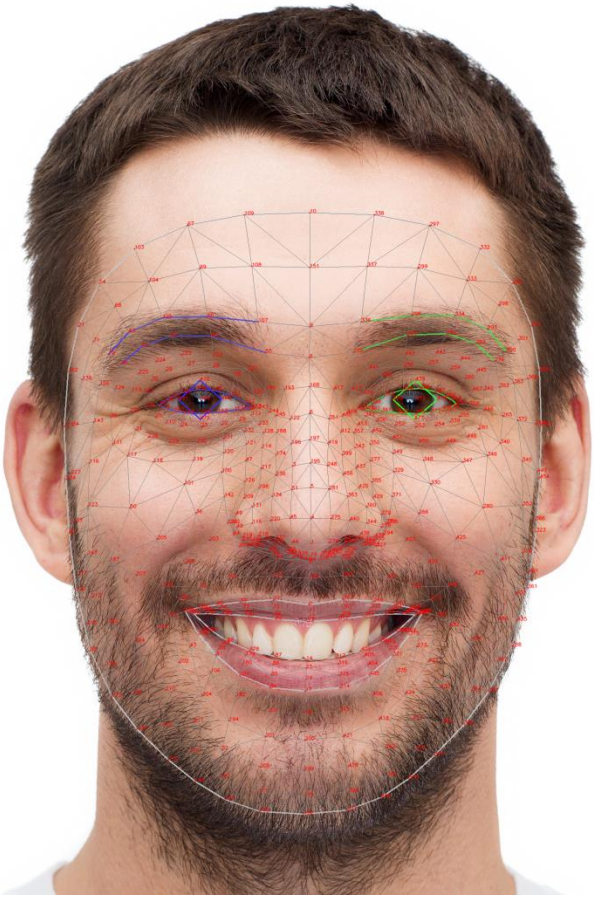


Fig 1- MediaPipe Detection

PyautoGUI Library:

PyautoGUI library is an automation library that allows mouse and keyboard control. Or we can say that it facilitates us to automate the movement of the mouse and keyboard to establish the interaction with the other application using the Python script. It provides many features, and a few are given below.

- We can move the mouse and click in the other applications' window.
- We can send the keystrokes to the other applications. For example - filling out the form, typing the search query to browser, etc.
- We can also take snapshots and give an image.
- It allows us to locate a window of the application, and move, maximize, minimize, resizes, or close it.
- Display alert and message boxes.

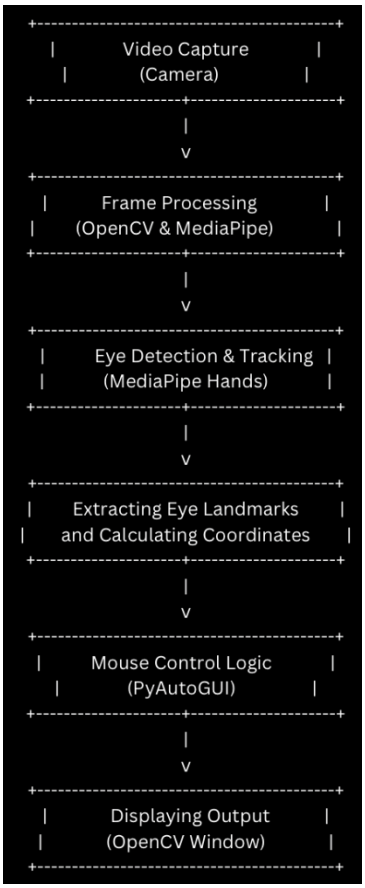


Fig. 2 – Process Of Vision Mouse

CHAPTER 2

REQUIREMENT SPECIFICATION

Software Requirements:

1. Python 3.12.2
2. OpenCV library is used for capturing video frames, image processing, and displaying images. Make sure to install OpenCV (cv2) library.
3. MediaPipe library is utilized for hand detection and tracking. Install the MediaPipe library (mediapipe) using pip.
4. PyAutoGUI library is used for simulating mouse movements and clicks. Install PyAutoGUI library using pip.

Hardware Requirements:

RAM: 4GB of RAM or More.

1. The system requires a webcam to capture real-time video frames.

2. A computer system capable of running Python scripts and handling real-time video processing.

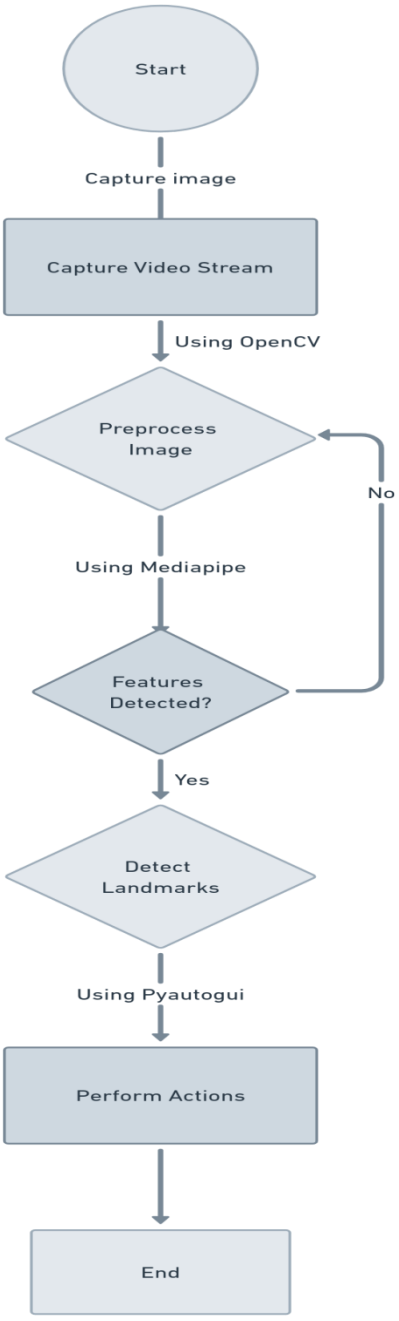
Input Requirements:

Eye movement: The system requires users to perform specific eye movement to control the virtual mouse. These gestures include closing the index finger and thumb for mouse clicks and moving the hand for cursor movement.

Screen Size: The system automatically detects the screen size using PyAutoGUI .Ensure that the screen resolution is accurately detected to perform mouse movements and clicks accurately.

CHAPTER 3
DESIGN

Flowchart of Vision Mouse :



CHAPTER 4
CODING

Project coding :

```
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)
```

Steps For Program:

1. Initialize the video capture using cv2.VideoCapture(0) to access the default camera.
2. Utilize the MediaPipe Face Mesh module (mp.solutions.face_mesh.FaceMesh) to detect facial landmarks in the captured frames.
3. Retrieve the screen dimensions using pyautogui.size() to calculate the cursor position on the screen.
4. Enter a while loop to continuously process video frames.
5. Read a frame from the video capture using cam.read().
6. Flip the frame horizontally using cv2.flip(frame, 1) for correct orientation.
7. Convert the frame to RGB color space using cv2.cvtColor(frame, cv2.COLOR_BGR2RGB).
8. Process the RGB frame with the Face Mesh model using face_mesh.process(rgb_frame).
9. Retrieve the facial landmark points from the output using output.multi_face_landmarks.
10. Calculate the screen coordinates based on the position of specific facial landmarks.
11. Move the cursor to the calculated screen coordinates using pyautogui.moveTo(screen_x, screen_y).
12. Detect left eye blinks by comparing the vertical positions of specific landmarks.
13. If the difference in vertical positions falls below a threshold, simulate a mouse click using pyautogui.click().
14. Show the processed frame with overlaid facial landmarks and eye blink detection using cv2.imshow().
15. Wait for a key press using cv2.waitKey(1).

CHAPTER 5

TESTING

1.Normal Lighting vs. Low Lighting Conditions:

Description: Test the application under normal lighting conditions and then under low lighting conditions to assess performance in different lighting environments.

Procedure: Place the webcam in an area with adequate lighting for the normal lighting condition test. Then, reduce the lighting or test the application in dimly lit conditions for the low lighting condition test.

Expected Outcome: The application should accurately detect eye movement and translate them into mouse movements and clicks in both normal and low lighting conditions.

2.Different User Distances from the Webcam:

Description: Test the application with users positioned at varying distances from the webcam to evaluate performance at different distances.

Procedure: Have users interact with the application while positioned at different distances from the webcam, ranging from close proximity to a moderate distance.

Expected Outcome: The application should accurately detect eye movement and translate them into mouse movements and clicks regardless of the user's distance from the webcam.

3.Testing for Accuracy in Cursor Movement and Click Simulation:

Description: Evaluate the accuracy of cursor movement and click simulation by performing precise eye movements .

Procedure: Perform various eye movement, including moving the eye to control cursor movement and closing the eye for click simulation.

Expected Outcome: The application should accurately track eye movements and close, resulting in smooth and precise cursor movement and click simulation.

4.Results and Analysis:

The testing outcomes will provide insights into the performance of the application under different conditions. The analysis will include:

5.Performance in Different Lighting Conditions:

Assess whether the application maintains functionality and accuracy in both normal and low lighting conditions.

Identify any potential limitations or issues related to lighting conditions affecting eye detection and tracking.

6.Effectiveness at Different User Distances:

Determine whether the application reliably detects eye movement and translates them into mouse actions regardless of the user's distance from the webcam.

Evaluate any discrepancies or challenges encountered at varying distances and propose potential solutions or optimizations.

7.Accuracy in Cursor Movement and Click Simulation:

Analyze the accuracy and responsiveness of cursor movement and click simulation in accordance with user eye movement.

Identify any instances of lag, imprecision, or misinterpretation of eye movement and propose refinements to improve performance.

CHAPTER 6

PROJECT EXECUTION (Screenshots)

```
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()
```

Fig. 3 – Importing OpenCV , MediaPipe and PyAutoGUI

```
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
```

Fig. 4 – Frame Detection

```
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))
```

Fig 5 – Landmarks Points

```
        if (left[0].y - left[1].y) < 0.004:
            pyautogui.click()
            pyautogui.sleep(1)
    cv2.imshow('Eye Controlled Mouse', frame)
    cv2.waitKey(1)
```

Fig 6- Eye Detection And Eye Control

CHAPTER 7

CONCLUSION

In conclusion, the virtual mouse system developed in this project signifies a groundbreaking advancement in assistive technology, particularly for individuals with physical disabilities. By harnessing the power of eye movement captured through a webcam, this system offers a revolutionary hands-free solution for computer interaction. The capability to control the mouse cursor with precision and ease empowers users to navigate digital environments with newfound independence and efficiency. The system's innovative approach to accessibility addresses a critical need in the realm of technology, paving the way for greater inclusivity. By eliminating the reliance on traditional input devices, such as keyboards and mice, individuals with physical disabilities can overcome barriers that previously hindered their full participation in digital activities. This breakthrough holds immense potential to enhance the quality of life for countless individuals by providing them with the tools to engage more fully in various aspects of daily life, including communication, education, work, and entertainment.

Moreover, the virtual mouse system's promising functionality and usability mark a significant leap forward in the quest for greater accessibility and inclusivity in technology. Its intuitive interface and seamless integration of hand tracking and mouse control ensure a user-friendly experience that caters to the diverse needs and preferences of individuals with physical disabilities. Looking ahead, as technology continues to evolve, the virtual mouse system stands poised to play a pivotal role in shaping the future of assistive technology. With ongoing advancements and refinements, this innovative solution holds the promise of further transforming the lives of individuals with physical disabilities, fostering greater independence, autonomy, and inclusion in the digital age.

In essence, the virtual mouse system represents more than just a technological innovation; it embodies a profound commitment to breaking down barriers, empowering individuals, and fostering a more inclusive society where everyone can harness the benefits of technology to lead fulfilling and meaningful lives.

CHAPTER 8

FUTURE WORK

1. Enhanced Gesture Recognition: Continuously refine gesture recognition algorithms to expand support for a wider range of eye movement and actions. This could involve incorporating machine learning techniques to improve accuracy and robustness in detecting and interpreting various hand movements.

2. Advanced Customization: Explore opportunities to expand customization options within the system, allowing users to personalize their experience further. This may include additional parameters for sensitivity, gesture mapping, and interface customization to accommodate diverse user preferences and needs.

3. Integration with Assistive Technologies: Investigate possibilities for integrating the virtual mouse system with other assistive technologies and devices. This integration could enhance accessibility and functionality by leveraging complementary technologies such as speech recognition, switch access, or brain-computer interfaces to provide alternative input methods and expand the system's capabilities.

4. User Feedback and Iterative Improvements: Implement a robust feedback mechanism to gather user input and insights continually. User feedback should inform iterative improvements to the system's design and implementation, addressing usability issues, refining features, and enhancing overall performance based on real-world usage scenarios.

REFERENCES

1. https://shorturl.at/isDGR
2. https://docs.opencv.org/4.x/d1/dfb/intro.html
3. https://developers.google.com/mediapipe/
4. https://pyautogui.readthedocs.io/en/latest/