VIRTUAL MOUSE CONTROLLED USING EYE GESTURES

A MINI PROJECT REPORT

Submitted By

MARIA ROBIN ANDREW - 211420104156

BALAMANI B – 211420104033

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

In

***C*omputer Science and Engineering**

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

2022 – 2023

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this project report **"Virtual Mouse using Eye Gestures"** is the bonafide work of **Maria Robin Andrew (211420104156) & Balamani (211420104033)** who carried out the project work under my supervision.

Signature	**Signature**
Dr.L.JABASHEELA , M.E .,Ph.D **PROFESSOR,** **HEAD OF THE DEPARTMENT**	**Dr.VALARMATHI .K M.E., Ph.D** **PROJECT GUIDE,** **PROFESSOR,**
DEPARTMENT OF CSE PANIMALAR ENGINEERING COLLEGE, NASARATHPETTAI, POONAMALLEE, CCHENNAI-600 123.	DEPARTMENT OF CSE PANIMALAR ENGINEERING COLLEGE, NASARATHPETTAI, POONAMALLEE, CCHENNAI-600 123.

Certified that the above candidate(s) were examined in the Mini Project Viva-Voce Examination held on...........................

DECLARATION BY THE STUDENTS

We **MARIA ROBIN ANDREW (211420104156), BALAMANI B (211420104033)** hereby declare that this project report titled "**VIRTUAL MOUSE USING EYE GESTURES**", under the guidance of **Mrs. VALARMATHI K** is the original work done by us and we have not plagiarized or submitted to any other degree in any university by us.

ACKNOWLEDGEMENT

We would like to express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D**. for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We express our sincere thanks to our beloved Directors **Tmt.C.VIJAYARAJESWARI, Dr.C.SAKTHI KUMAR, M.E.,Ph.D** and **Dr.SARANYASREE SAKTHI KUMAR B.E.,M.B.A.,Ph.D.**, for providing us with the necessary facilities to undertake this project.

We also express our gratitude to our Principal **Dr.K.MANI, M.E., Ph.D.** who facilitated us in completing the project.

We thank the Head of the CSE Department, **Dr. L.JABASHEELA, M.E.,Ph.D.**, for the support extended throughout the project.

We would like to thank our parents, friends, project Guide **Dr.VALARMATHI.K M.E., Ph.D**., and coordinator **Dr.HEMLATHA DHEVI. A M.E., Ph.D.**, and all the faculty members of the Department of CSE for their advice and encouragement for the successful completion of the project.

Name Of The Students

Maria Robin Andrew (211420104156)

Balamani B (211420104033)

ABSTRACT

The usage of computers has been increased day to day, yet the physically disabled people find it near to impossible to use them. Controlling a mouse which is the simple GUI interaction can be tough job for the challenged persons. To solve this issue, we propose a virtual controlled mouse which can help them to fill the gap. For those who find difficulty in touchscreens and physical mouse inaccessibility, this virtual mouse helps them to use mouse controls using eye gaze technique which makes mouse controls using eye movements. Eye gaze tracing a human eye for inputs. Eye movements and speech commands are one of the most reliable interactions for the physically disabled people who cannot make use of their hands. Our project is concentrated on the using the eye movements and our virtual mouse is focused on these eye movement which can be helpful for mouse control by these people. Eye movements are regarded as inputs in many software for tracking to analysis purposes. The controls of this virtual mouse are simple eye gestures such as left eye blink for a left click and right eye blink for a right click. The eye gaze gets captured in real time frames from a web camera connected to the system without any other external hardware requirements. The virtual mouse moves the mouse cursor to the targeted coordinates according to the right eye position captured by the camera or a webcam for every frame. Whenever a mouse action is performed, the project identifies it and performs the respective action. This project is simpler to use and a user-friendly HCI and it will help to break the barrier for the usage of computers by physically challenged people.

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

1.1 OVERVIEW

Computers and technology have become integral parts of our day-to-day lives, constantly evolving and changing the way we live, work, and communicate with one another. With each passing day, new technologies are being developed, and existing ones are being improved upon to make them more user-friendly, efficient, and accessible to all. As a result, the way we use computers and other technological devices is constantly changing and adapting to new trends and advancements. They may have difficulty typing, using a mouse, or performing other tasks that require fine motor control. This can make it challenging to complete work tasks, communicate with others, or even engage in leisure activities. This paper proposes an AI Virtual mouse which is controlled by the eye gestures to overcome the challenge of physical usage of the mouse. Eye gestures and face movements can be captured by a video capturing devices, which are used in human-computer interaction. The virtual mouse is operated by capturing the right eye position respective to the captured frame for movement of the cursor. The eye gestures such as blinks are used for mouse controls.

The design of the virtual mouse is made simple in terms of the interface and the interaction. This simple design of the mouse makes it upgradable and enhancement is made easier. An external hardware may not be required since the virtual mouse is a software. the only extra hardware required is a video capturing device like a webcam.

The software is built using in python platform which is supported in all operating systems of the computer or can be installed. The python package, OpenCV which is the library for computer vision is used in the virtual mouse software. In the proposed system, the model makes use of the MediaPipe package for facial landmark detection which is used for identification of eye gestures and eye gazing. PyAutoGUI, another python package is used to for mouse movement in the screen from the interactions recieved from the users. Finally, the virtual mouse is quite promising for its usage and proposed problem and the results are found to be at descent levels without much resources required.

2.2 PROBLEM DEFINITION

- The main objective of this project is to design a virtual mouse which should be useful for physically challenged people who cannot operated the mouse physically and should be able to operate using the eye gestures.

- Can also used to overcome problems in the real world such as situations where there is no space to use a physical mouse and also for the persons who have problems in their hands and are not able to control a physical mouse.

CHAPTER 2
SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

- Some companies during Covid-19 use virtual mouse with eye gaze controls to reduce the spread of the pandemic disease in an onsite workspaces. Third party companies developed these software. The companies which use these will not much work since it is not efficient for fit human being to control and work using the eye for a longer time.

- Some popular eye-tracking devices include Tobii Eye Tracker 5, EyeTech VT3 Mini, Pupil Labs, and EyeTribe. These devices use infrared cameras to track the user's eye movements and translate them into mouse movements on the screen.

- There are also software solutions available, such as Windows Control by Tobii Dynavox and OptiKey, which allow users to control their computer using eye-tracking technology.

- Additionally, virtual eye gaze mouse implementation is becoming more common in everyday devices, such as smartphones and tablets. For example, the latest versions of Apple's iOS and iPadOS include built-in eye-tracking features that allow users to control their devices through eye movements.

DISADVANTAGES:

The above-mentioned systems carried out on virtual mouse using eye gestures are on open source and may require more resources utilisation. Some are not upgradable or inconsistent for updates. Most of these models require high computation and complexity are high. The modules used are high which require a lot of space to store. These models are not able to be used for research since there not open-sourced projects.

2.2 PROPOSED SYSTEM

The python module OpenCV is used for computer vision; Mediapipe module is for capturing facial landmark and PyAutoGUI is used for mouse movements. The system makes use of Artificial Intelligence – Deep learning concepts overall for tracking and recognising the eye gestures. This Mediapipe package is not mostly used in any software which are existing so far in this type of project model.

FEATURES OF THE PROPOSED SYSTEM

- Provides greater flexibility for the usage and upgradation of the software.
- Easier to adapt to any operating systems.
- Uses less computation and resources for utility of the software.
- Less physical usage for mouse movements.
- More fun and entertaining to use the system with easy mouse controls.

2.3 DEVELOPMENT ENVIRONMENT

SOFTWARE REQUIREMENTS

- Window 8 or higher
- Python 3.8
- OpenCV module
- Mediapipe module
- Pyautogui module
- Webcam drivers

HARDWARE REQUIREMENTS

- Processor: minimum 1.8 GHz clock speed
- RAM: 4 GB
- Peripheral webcam at least 30 frames/second

CHAPTER 3
SYSTEM DESIGN

3.1 UML DIAGRAMS

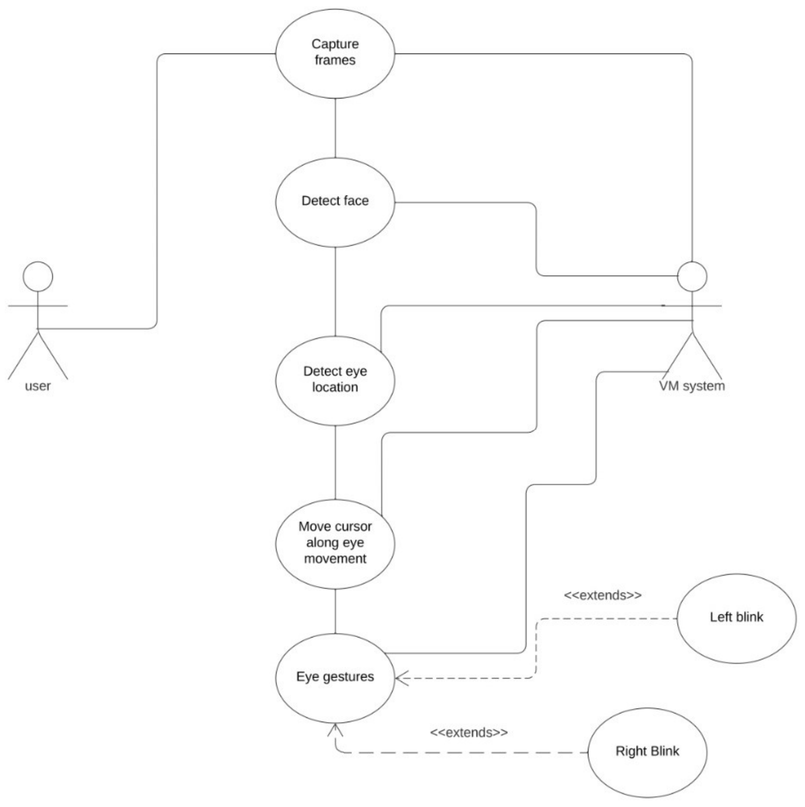

Fig 3.1.1 Use case diagram

Fig 3.1.1 shows the use case diagram of the virtual mouse system interactions between the user and the system.

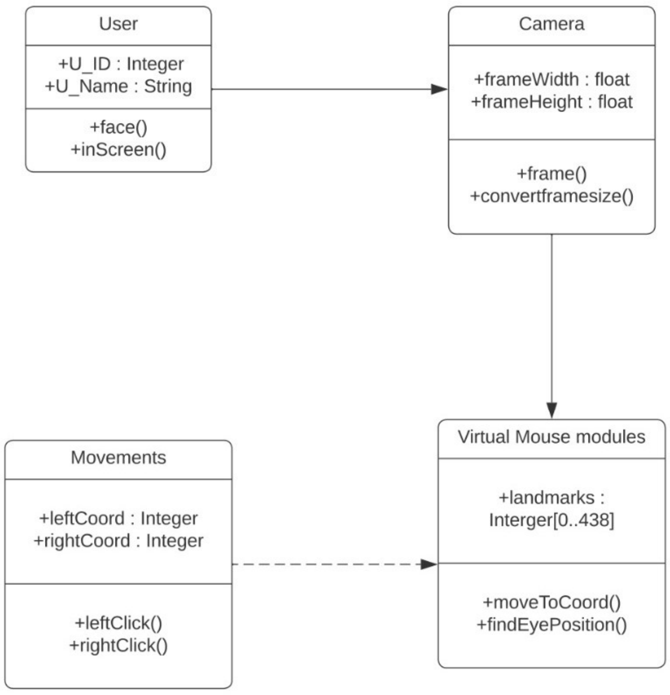

Fig 3.1.2 Class diagram

Fig 3.1.2 shows the class diagram of the virtual mouse system. It shows the classes mainly used in the system.

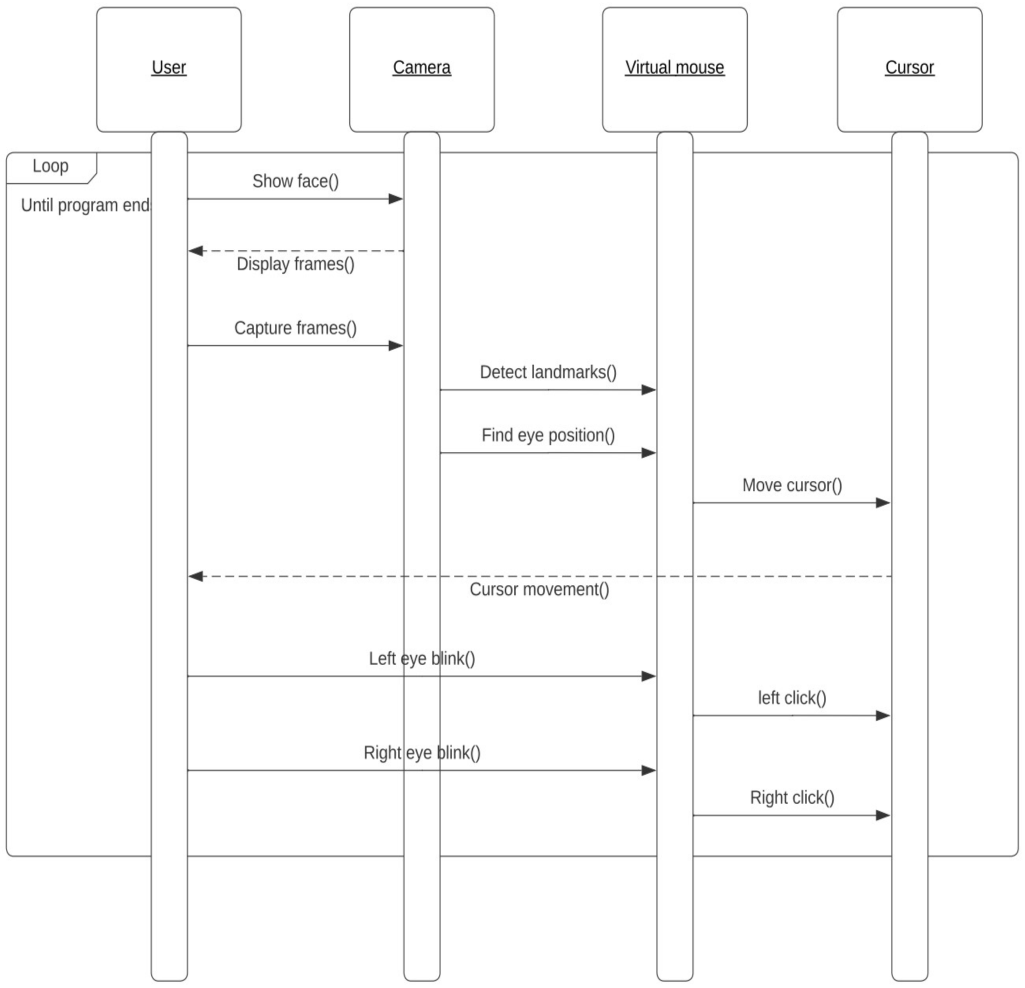

Fig 3.1.3 Sequence diagram

Fig 3.1.3 shows the sequence diagram which shows the interaction sequence between the different modules and actors.

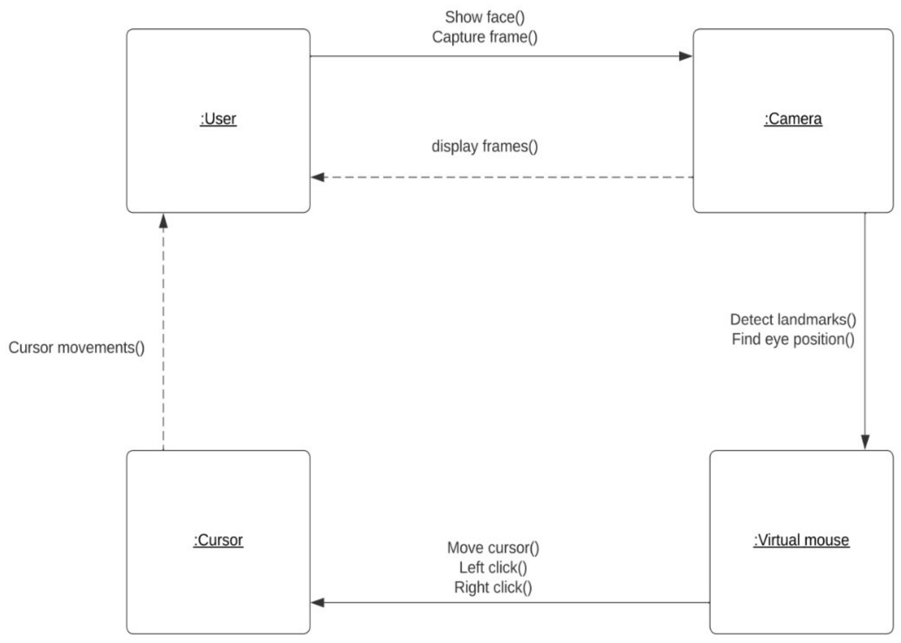

Fig 3.1.4 Collaboration diagram

Fig 3.1.4 depicts the collaboration diagram which shows the interactions between the class modules.

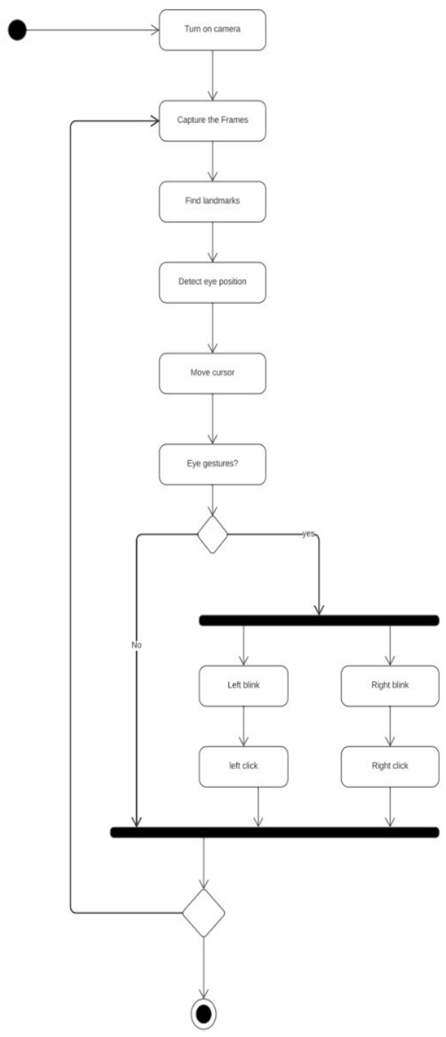

Fig 3.1.5 Activity diagram

Fig 3.1.5 shows the activity sequence of the virtual mouse. It shows the activities performed by the virtual mouse system in the sequence.

3.2 DATA DICTIONARY

This is normally represented as the data about data. It is also termed as metadata some times which gives the data about the data stored in the database. It defines each data term encountered during the analysis and design of a new system. Data elements can describe files or the processes.

In this system, the mediapipe library consists of the landmarks data of the face.

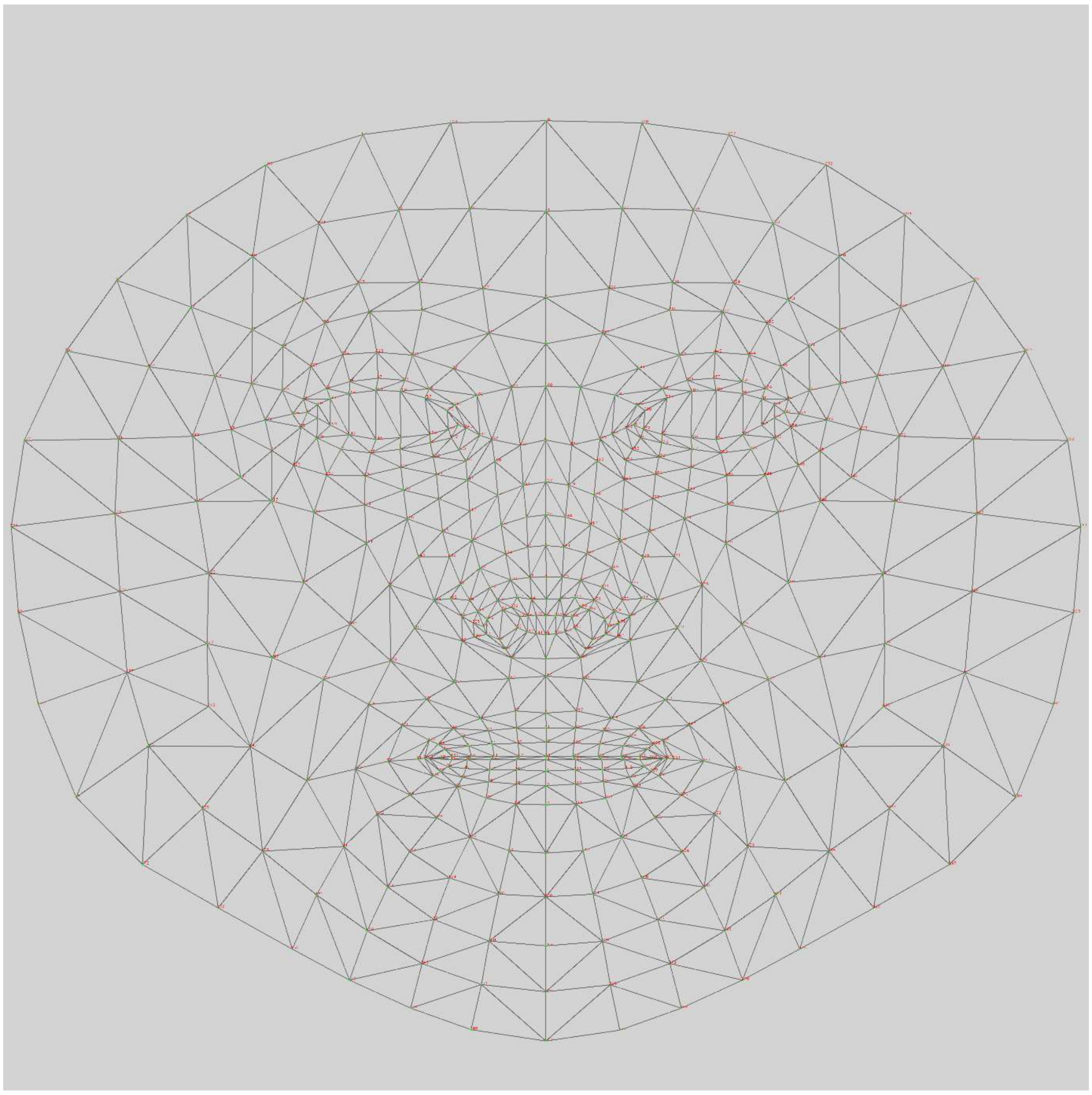
Fig 3.2.1 canonical face model UV visualization

Fig 3.2.1 shows the canonical face model with landmarks. There consist of 468 landmarks which helps us to identify the facial movements. The mediapipe library helps us to identify these landmarks.

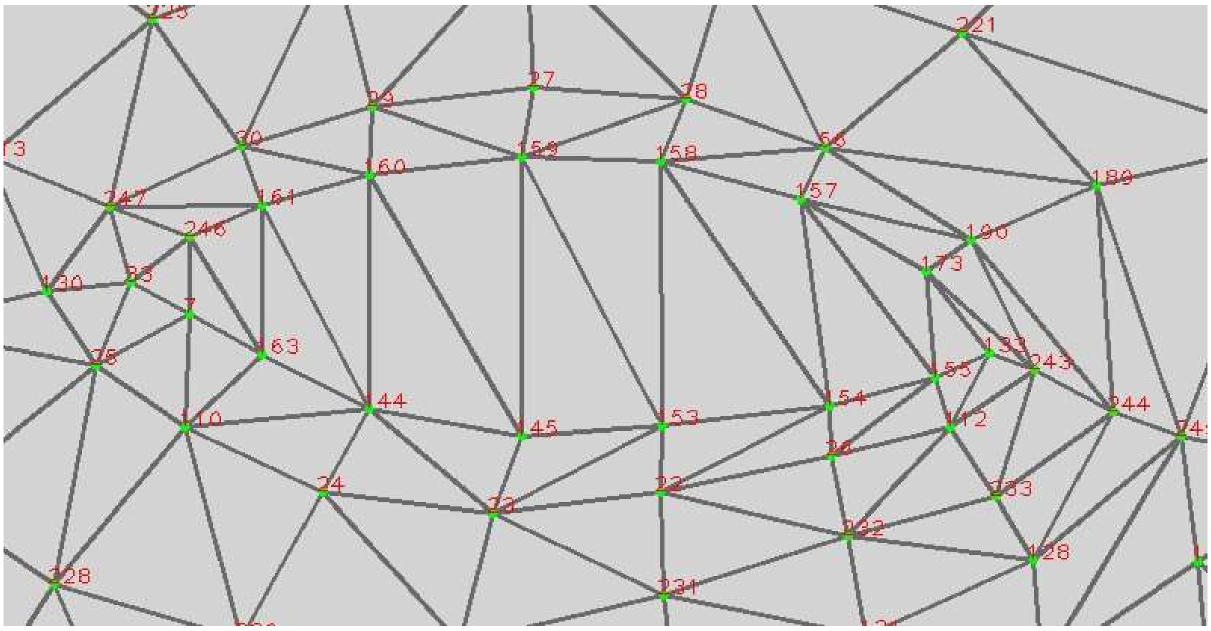

Fig 3.2.2 left eye landmarks

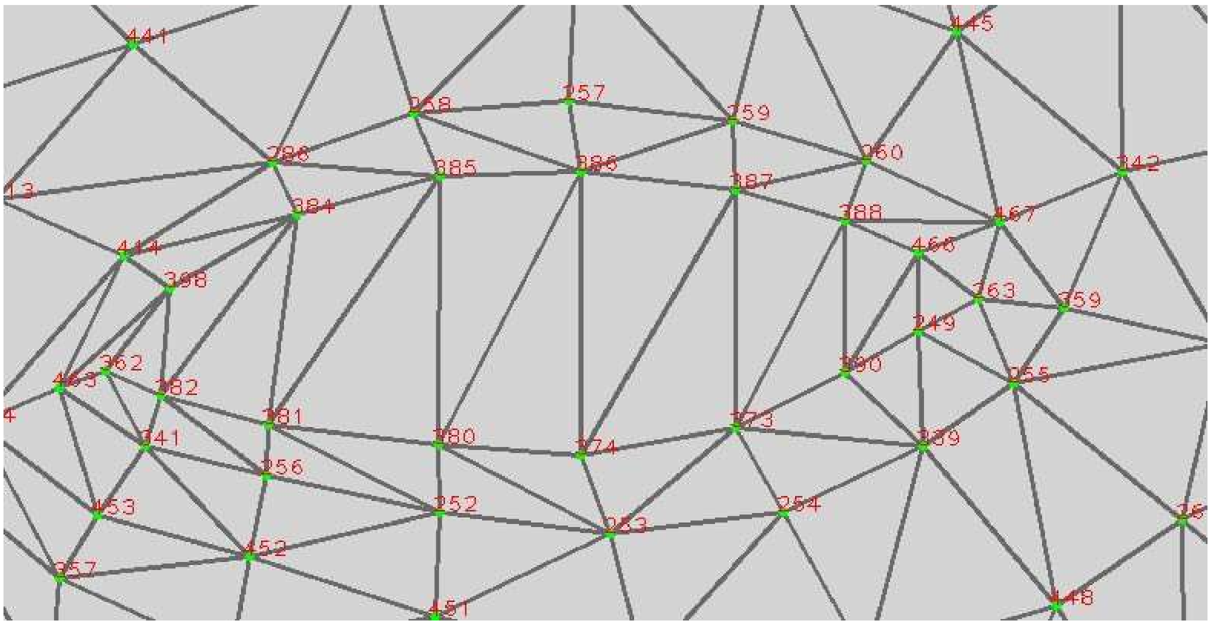

Fig 3.2.3 right eye landmarks

3.3 E-R DIAGRAM

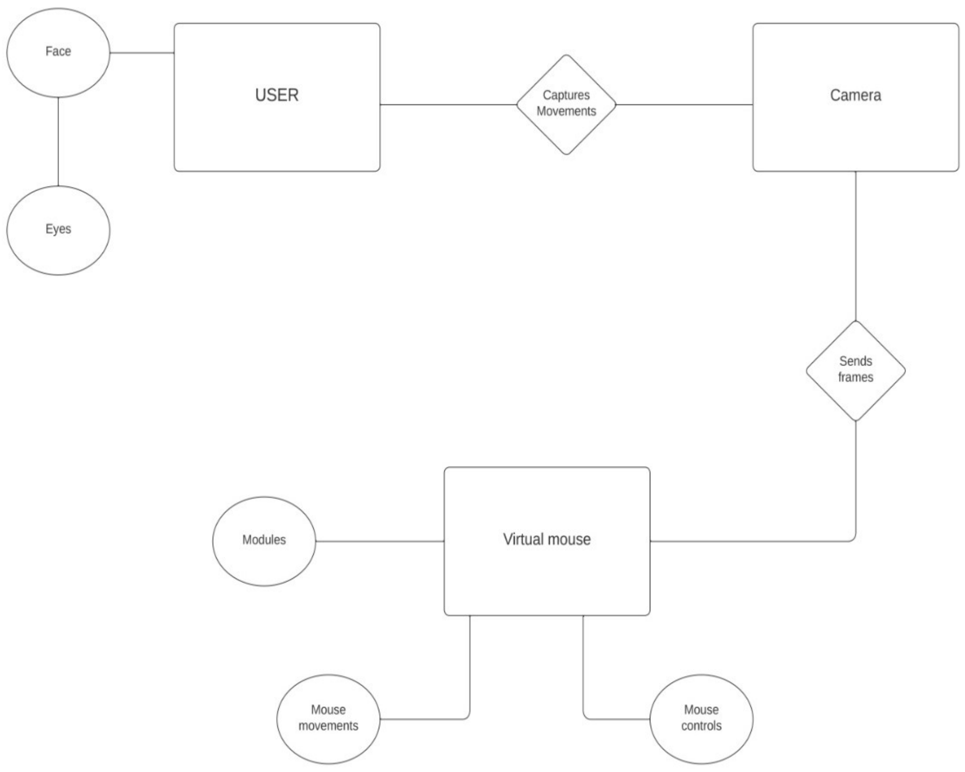

Fig 3.3.1 E-R Diagram

Figure 3.3.1 represents the Entity – Relationship diagram within the virtual mouse system.

3.4 DATA FLOW DIAGRAM

A data flow diagram (DFD) is a graphical representation of the “low of data through an information system, modelling its process aspects. Often they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs can also be used for the visualization of data processing.

3.4.1 LEVEL – 0 DFD

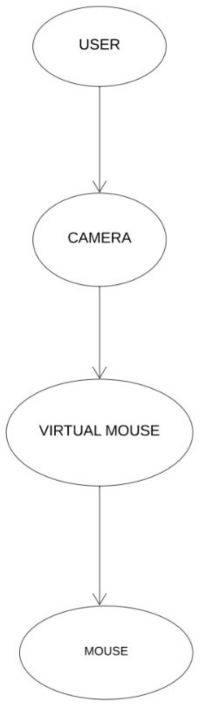

Fig 3.4.1.1 Level- 0 Data flow diagram

Figure 3.4.1.1 shows the Level – 0 Data Flow Diagram which shows the basic data flow in the virtual mouse system.

3.4.2 LEVEL – 1 DFD

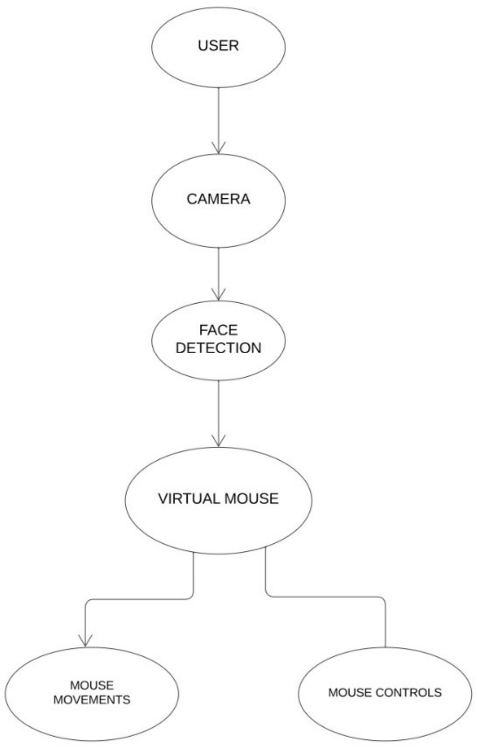

Fig 3.4.2.1 Level – 1 Data flow diagram

Figure 3.4.2.1 shows the Level – 1 Data flow diagram which shows the data flow of the virtual mouse in the components level.

3.4.3 LEVEL – 2 DFD

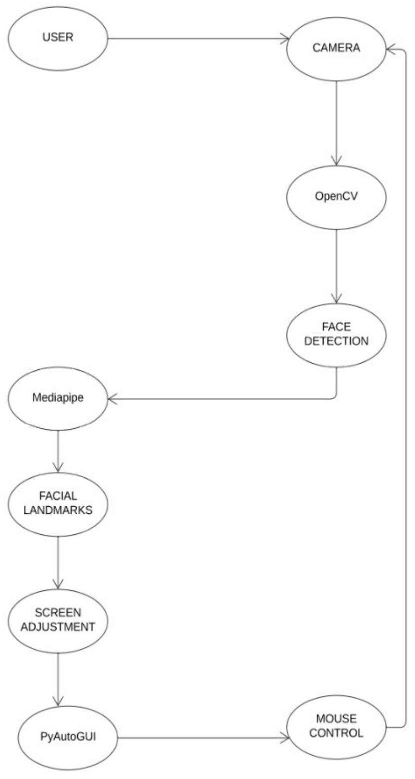

Fig 3.4.3.1 Level – 2 Data flow diagram

Figure 3.4.3.1 depicts level – 2 data flow diagram which shows module level data flow in the virtual mouse system.

CHAPTER 4
SYSTEM ARCHITECTURE

4.1 ARCHITECTURE OVERVIEW

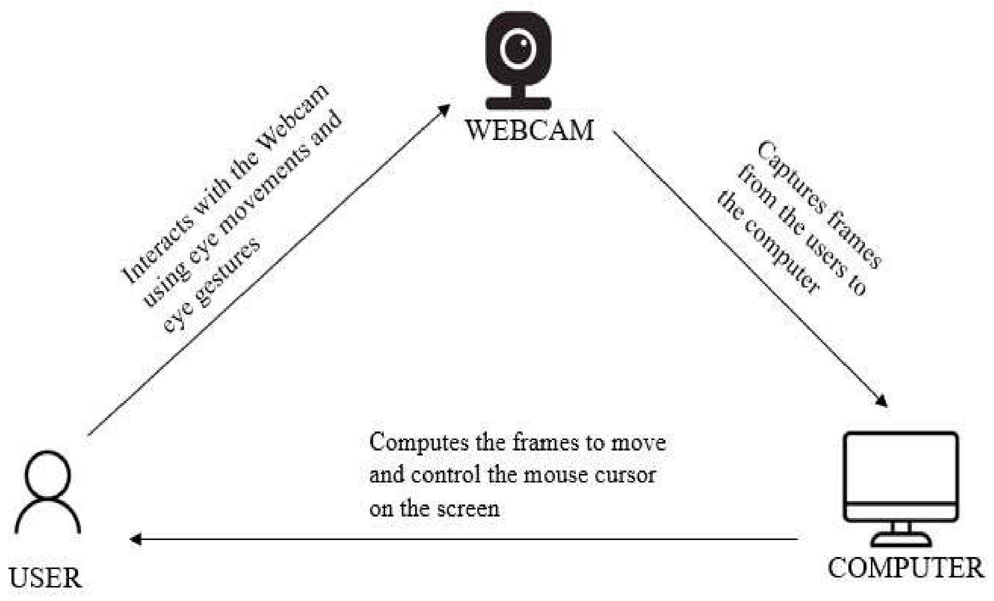
WEBCAM
Interacts with the Webcam using eye movements and eye gestures
Captures frames from the users to the computer
Computes the frames to move and control the mouse cursor on the screen
USER
COMPUTER

Fig 4.1.1 Architecture diagram for Virtual Mouse system

The figure 4.1.1 shows the basic architecture diagram of the virtual mouse system. It depicts the overall workflow of the Virtual mouse system. This architecture can be also considered as the human-computer interaction architecture. Initially, the user interacts with the computer via the webcam. The webcam sends the frames to the computer. The virtual mouse system presented in the computer computes with the frames and gives the output in the screen which can be seen by the user.

The computations done by the virtual mouse system can be seen the architecture diagram in the figure 4.1.2

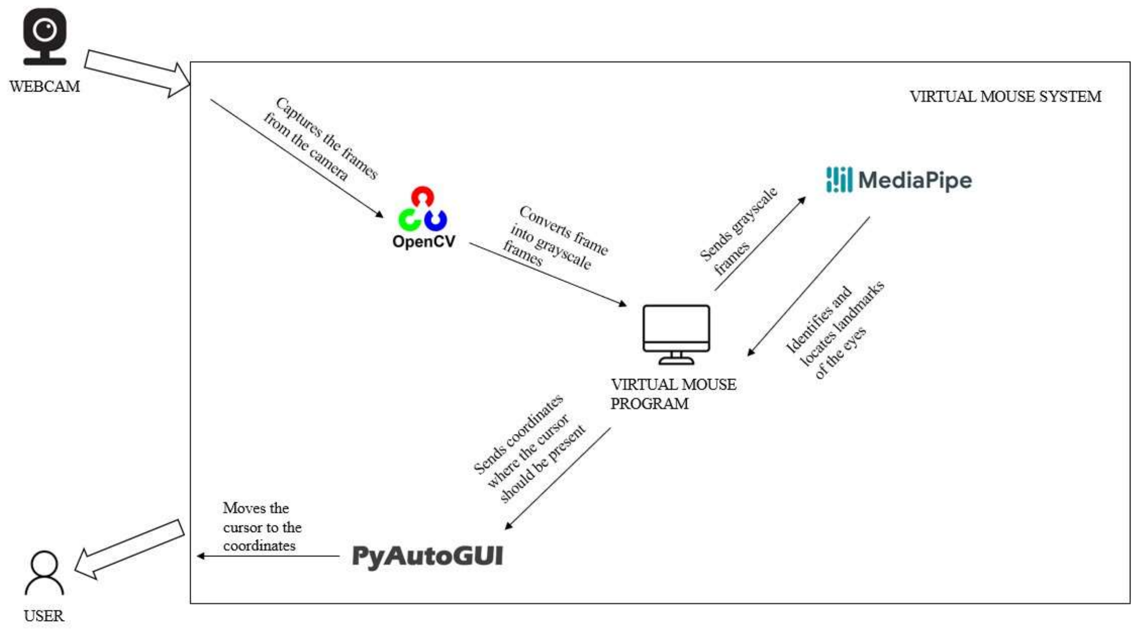
WEBCAM
VIRTUAL MOUSE SYSTEM
Captures the frames from the camera
OpenCV
Converts frame into grayscale frames
Sends grayscale frames
MediaPipe
Identifies and locates landmarks of the eyes
VIRTUAL MOUSE PROGRAM
Sends coordinates where the cursor should be present
PyAutoGUI
Moves the cursor to the coordinates
USER

Fig 4.1.2 Virtual mouse module architecture

The virtual mouse system receives the frames via the OpenCV module. The OpenCV module converts the received frames from a coloured frame to a greyscale frame since greyscale frames are easier for further processing. Then, the mediapipe module identifies the landmarks of the face from these modules. Since mediapipe identifies only one face, if there are more than one face, the first clear face would be recognised. After the landmarks are plotted, the system takes the required landmarks (right eye and left eye landmarks). Using OpenCV module, the coordinates of the landmarks is changes accordingly to the computer system's screen resolutions. The mouse cursor is positioned according to the coordinated of the right eye received. The eye gesture for the left click and right click of the mouse is checked by the landmark processing.

4.2 MODULE DESCRIPTION

For the purpose of detection of facial landmarks and eye gestures, the MediaPipe framework [0.9.1.0] is used, and OpenCV library [4.7.0.72] is used for computer vision. The algorithm makes use of the machine learning and deep learning concepts for facial recognition and landmark identification. The PyAutoGUI module is used for GUI control of the mouse cursor.

MEDIAPIPE

MediaPipe is a comprehensive, cross-platform, and open-source framework developed by Google that enables developers to build complex multimedia processing pipelines. It consists of a set of pre-built models and functions that can be customized to create real-time multimedia applications for desktop, mobile, and edge devices.

MediaPipe's modular architecture allows developers to assemble a pipeline using pre-built components, or build their own components from scratch. The framework supports a wide range of media formats, including video, audio, and sensor data, and provides functionality for tasks such as object detection, facial recognition, gesture recognition, and pose estimation. MediaPipe leverages machine learning techniques, such as neural networks, to deliver highly accurate results in real-time.

The framework supports popular machine learning frameworks like TensorFlow and provides tools for training custom models. MediaPipe is designed to be easily integrated with other frameworks and libraries, such as OpenCV and OpenGL, and supports a variety of programming languages, including C++, Python, and Java.

MediaPipe's flexibility, scalability, and modularity make it a powerful tool for building real-time multimedia applications. Its pre-built models and functions allow developers to quickly prototype and build complex pipelines, while its support for custom components and machine learning models enables developers to create highly specialized applications that meet their specific needs. Additionally, its open-source nature allows for community contributions, making it an ever-evolving framework.

OPENCV

OpenCV (Open Source Computer Vision Library) is a widely used open-source computer vision and machine learning library, developed by Intel and now maintained by the OpenCV.org community. It provides a comprehensive set of tools and functions for image and video processing, including object detection and tracking, face recognition, optical flow, stereo vision, and more.

OpenCV is written in C++, but also provides APIs for Python, Java, and other programming languages, making it accessible to a wide range of developers. It runs on multiple platforms, including Windows, Linux, macOS, Android, and iOS. One of the strengths of OpenCV is its efficient implementation of algorithms, making it suitable for real-time and embedded applications. The library also supports GPU acceleration through OpenCL and CUDA, providing significant speedups for computationally intensive tasks.

OpenCV has a large and active community, contributing to its continuous development and evolution. Its modular design allows developers to easily extend its capabilities, and it integrates well with other libraries, such as NumPy and Matplotlib.

Overall, OpenCV is a powerful and versatile tool for computer vision and machine learning applications, widely used in industry and academia. Its broad range of functions, high performance, and open-source nature make it a popular choice for developers working in the fields of robotics, autonomous vehicles, surveillance, and more.

PYAUTOGUI

PyAutoGUI is an open-source cross-platform Python library used for automating keyboard and mouse functions. It allows users to programmatically control the keyboard and mouse, as well as capture and manipulate screenshots. PyAutoGUI is often used for tasks such as GUI automation, testing, and repetitive tasks that require a large amount of mouse and keyboard input.

PyAutoGUI works on various platforms, including Windows, macOS, and Linux. It provides functions for simulating keyboard keys, mouse clicks, and scrolling, as well as functions for moving the mouse and capturing screenshots. The library also provides functions for controlling the delay between actions, making it possible to accurately simulate human interaction with a computer.

PyAutoGUI's key strength lies in its simplicity and ease of use. It offers a straightforward interface and requires minimal setup to get started. Additionally, it is highly customizable, with the ability to define custom keyboard shortcuts and mouse movements.

One of the key benefits of PyAutoGUI is its ability to automate tasks that would otherwise be tedious or time-consuming. This includes tasks such as filling out forms, data entry, and GUI testing. With its ability to accurately simulate human input, PyAutoGUI provides a reliable way to automate these tasks, saving time and increasing productivity.

CHAPTER 5
SYSTEM IMPLEMENTATION

5.1 CODING

main.py

```
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]]
        right = [landmarks[374], landmarks[386]]
        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))
        for landmark in left:
            x = int(landmark.x * frame_w)
            y = int(landmark.y * frame_h)
        if (left[0].y - left[1].y) < 0.009:
            if(right[0].y - right[1].y) > 0.009:
                pyautogui.click()
                pyautogui.sleep(1)
        elif (right[0].y - right[1].y) < 0.009:
            pyautogui.click(button="right")
            pyautogui.sleep(1)
    cv2.imshow('Eye Controlled Mouse', frame)
    cv2.waitKey(1)
```

requirements.txt

```
MouseInfo   0.1.3
Pillow      9.4.0
```

PyAutoGUI	0.9.53
PyGetWindow	0.0.9
PyMsgBox	1.0.9
PyRect	0.2.0
PyScreeze	0.1.28
absl-py	1.4.0
attrs	22.2.0
contourpy	1.0.7
cycler	0.11.0
flatbuffers	23.3.3
fonttools	4.39.0
importlib-resources	5.12.0
kiwisolver	1.4.4
matplotlib	3.7.1
mediapipe	0.9.1.0
numpy	1.24.2
opencv-contrib-python	4.7.0.72
packaging	23.0
pip	22.3.1
protobuf	3.20.3
pyparsing	3.0.9
pyperclip	1.8.2
python-dateutil	2.8.2
pytweening	1.0.4
setuptools	65.5.1
six	1.16.0
wheel	0.38.4
zipp	3.15.0

CHAPTER 6
SYSTEM TESTING

6.1 TESTCASE AND REPORTS

TEST CASE ID	TESTCASE/ ACTION TO BE PERFORMED	EXPECTED RESULT	ACTUAL RESULT	PASS/ FAIL
1	User eye constantly blinking	It should detect eye	Not Detected	FAIL
2	User eye is dirty	It should detect eye	Detected	PASS
3	User having shivering	It should detect eye	Detected	PASS
4	User specs not detect	It should detect eye	Detected	PASS
5	Eye not blink properly	It should do appropriate action	Action not performed	FAIL
6	Noisy Background	It should detect eye	Detected	PASS
7	User is 1m away from camera	It should detect eye	Detected	PASS

8	User is not rightly perpendicular to the camera	It should detect eye	Detected	PASS
9	Left eye blink	Left click should be performed	Left click performed	PASS
10	Right eye blink	Right click should be performed	Right click	PASS
11	Two face is seen	Should focus on one face	One face detected	PASS

CHAPTER 7
CONCULSION

7.1 CONCULSION

We have implemented a virtual mouse system to access the mouse cursor on the computer screen using only face movements. With the use of a camera and python technology, the system architecture is achieved. User is able to view eye movements and eye gestures captured through the camera which is displayed on the computer screen, accordingly the user can move the mouse cursor as needed and also perform various mouse actions. The proposed system is feature based thus allowing any user to use the system without prior registration. This system is especially useful for the people with physical disability of the hand usage and in workplaces where physical touch is hazardous or cannot be used. Thus, we implemented a basic upgradable model for a virtual mouse controlled using eye gestures.

7.2 FUTURE ENCHANEMENTS

We want to improve our system by adding timers for the usage of the system so that the user would be using the virtual mouse for a long period of time. We would also like to integrate our virtual mouse with keyboard so that the computer usage would look like completely virtually. The mouse can be improved by adding shortcut key such as volume control, brightness, windows, etc with new gestures.

CHAPTER 8
APPENDICES

CHAPTER 9
REFERENCES

- Nandini Modi and Jaiteg Singh "A comparative analysis of deep learning algorithms in eye gaze estimation" Published in: 2022 International Conference on Data Analytics for Business and Industry (ICDABI) View at: publisher site | IEEE.

- Arqam M. Al-Nuimi and Ghassan J. Mohammed "Face Direction Estimation based on Mediapipe Landmarks" Published in: 2021 7th International Conference on Contemporary Information Technology and Mathematics (ICCITM) View at: publisher site | IEEE.

- Victoria A. Sablina, Michael B. Nikiforov and Alexander V. Savin "Comparison of Facial Landmark Detection Methods for Micro-Expressions Analysis" Published in: 2021 10th Mediterranean Conference on Embedded Computing (MECO) View at: publisher site | IEEE.

- Reyhan Seher Baştuğ, Bartu Yeşilkaya, Mazlum Unay and Aydın Akan "Virtual Mouse Control by Webcam for the Disabled" Published in: 2018 Medical Technologies National Congress (TIPTEKNO) View at: publisher site | IEEE.

- Jhankar Mahbub CEO of Programming Hero View at: https://www.youtube.com/c/ProgrammingHero.