VIRTUAL MOUSE AND KEYBOARD USING GESTURES FOR RECOGNITION

A Project Stage - II Report

Submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

from

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD

By

SATHISH KUMAR.K (18C11A05C8)

SHIRIDI SAI.P (18C11A05C9)

SRIHARI.CH (18C11A05E4)

VINAY.B (18C11A05H4)

Under the guidance of

Mrs. M. SHAILAJA M.Tech

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

ANURAG ENGINEERING COLLEGE

AUTONOMOUS

(Affiliated to JNTU-Hyderabad & Approved by AICTE-New Delhi)

ANANTHAGIRI (V) (M), SURYAPETA (Dt), TELANGANA -508206.

www.anurag.ac.in

2021-2022

ANURAG ENGINEERING COLLEGE

AUTONOMOUS

(Affiliated to JNTU-Hyderabad & Approved by AICTE-New Delhi)
ANANTHAGIRI (V) (M), SURYAPETA (Dt), TELANGANA -508206.
www.anurag.ac.in

CERTIFICATE

This is to certify that the project stage - II work entitled " ***VIRTUAL MOUSE AND KEYBOARD USING GESTURES FOR RECOGNITION*** *" is a bonafide work done by "****SATHISH KUMAR.K (18C11A05C8), SHIRIDI SAI.P (18C11A05C9), SRIHARI.CH (18C11A05E4) & VINAY.B (18C11A05H4)****" in the partial fulfillment for the award of Bachelor of Technology in Computer Science & Engineering from JNTU, Hyderabad during the year* ***2021-2022****.*

This work has not been submitted to any other university or institute or organization for the award of any degree or diploma.

Mrs. M. SHAILAJA M.Tech
SUPERVISOR

Dr. M. MURUGESAN Ph.D., M.Tech
H.O.D., C.S.E

EXTERNAL EXAMINER

Dr. M.V. SIVA PRASAD Ph.D., M.Tech
PRINCIPAL

TRUEVOLTS RESEARCH CENTER(P),LTD

TO WHOM IT MAY CONCERN

This is to certify that ***Mr. SATHISH KUMAR.K (18C11A05C8), SHIRIDI SAI.P (18C11A05C9), SRIHARI.CH (18C11A05E4)*** *&* ***VINAY.B (18C11A05H4)*** *studying* ***Computer Science Engineering*** *in* ***Anurag Engineering College, Kodad*** *has successfully completed their Major project on* ***" Virtual Mouse and Keyboard using Gestures for Recognition "*** *in our organization for a period of 45 days. Their performance during this project period was satisfactory.*

We wish all the best in their future endeavors.

Thanks & Regards

Authorized signature

A6, 2nd FLOOR, EUREKA COURT, KS BAKERY BUILDING, OPP. R.S.BROTHERS LANE, AMEERPET,
HYDERABAD, TELANGANA - 500073.

www.truevolts.com +91 9908665239

ACKNOWLEDGEMENTS

This report will certainly not be completed without due acknowledgements paid to all those who helped us during this project stage - II work.

We would like to express our gratefulness to the project trainer **Mr. SRIKANTH** and **TRUEVOLTS RESEARCH CENTER (P), LTD** for their valuable guidance in carrying out this project stage - II work.

We express sincere thanks to our supervisor **Mrs. M. SHAILAJA,** for giving moral support, kind attention and valuable guidance throughout this project stage - II work.

It is our privilege to thank **Dr. M. MURUGESAN**, Head of the Department for his encouragement during the progress of this project stage - II work.

We are thankful to both Teaching and non-teaching staff members of **CSE DEPARTMENT** for their kind cooperation and all sorts of help in bringing out this project stage - II work successfully.

We derive great pleasure in expressing our sincere gratitude to the principal **Dr. M.V. SIVA PRASAD** for his timely suggestions, which helped us to complete this project stage - II work successfully.

We are thankful to the **ANURAG ENGINEERING COLLEGE** for providing required facilities during the project stage - II work.

We would like to thank our parents and friends for being supportive all the time, and we are very much obliged to them.

SATHISH KUMAR.K (18C11A05C8)
SHIRIDI SAI.P (18C11A05C9)
SRIHARI.CH (18C11A05E4)
VINAY.B (18C11A05H4)

VIRTUAL MOUSE AND KEYBOARD USING GESTURES FOR RECOGNITION

CONTENTS

ABSTRACT

Nowadays computer vision has reached its pinnacle, where a computer can identify its owner using a simple program of image processing. In this stage of development, people are using this vision in many aspects of day to day life, like Face Recognition, Color detection, Automatic car, etc. In this project, computer vision is used in creating an Optical mouse and keyboard using hand gestures. The camera of the computer will read the image of different gestures performed by a person's hand and according to the movement of the gestures the Mouse or the cursor of the computer will move, even perform right and left clicks using different gestures. Similarly, the keyboard functions may be used with some different gestures, like using one finger gesture for alphabet select and four-figure gesture to swipe left and right. It will act as a virtual mouse and keyboard with no wire or external devices. The only hardware aspect of the project is a web-cam and the coding is done on python using Anaconda platform. Here the Convex hull defects are first generated and then using the defect calculations an algorithm is generated and mapping the mouse and keyboard functions with the defects. Mapping a couple of them with the mouse and keyboard, the computer will understand the gesture shown by the user and act accordingly.

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

Figure	Description	Page

INTRODUCTION

Chapter-I

INTRODUCTION

The Computer webcam is capturing the video of the person sitting in front of the computer, there will be a small green box which will be generated in the middle of the screen. In that green box, the objects shown will be processed by the code and matched with it if it matches then a red colored border will be generated, which means the computer has identified the object and then by moving the object the mouse cursor can be moved. This will not only help in the security of the computer but also help in generating a virtual computational experience. Here in the place of different objects, using hand gestures one gesture will be moving the cursor, the different gesture will be used for right click and different for left click, similarly with a simple gesture can do the keyboard functions virtually that may have been done on some keyboard as a physical aspect. It the gesture does not match the box will show an only green box when the known gesture is observed a red border will occur.

A small green box will appear in the center of the screen when the computer webcam captures the footage of the person sitting in front of the computer. The objects displayed in that green box will be processed by the code and matched with it. If it matches, a red colored border will appear, indicating that the computer has recognized the item and that the mouse cursor can be moved by moving the object. This will aid not only in computer security but also in the construction of a virtual computational experience. In the place of various objects, hand gestures will be used to move the cursor, a different gesture will be used for right click, and a different gesture will be used for left click, and similarly, a simple gesture can do the keyboard functions virtually that may have been done on a physical keyboard. If the gesture does not match, the box will show simply a green border; however, if the known gesture is detected, the box will show a red border. Some hardware, such as the mouse, the dongle to connect to the PC, and a battery to power the mouse to function, are utilized when using a wireless or Bluetooth mouse, but in this article, the user uses his or her built-in camera or webcam and uses hand gestures to control the computer mouse operations. The web camera in the suggested system records and analyses the acquired frames, detects the various hand motions and hand tip gestures, and then performs the specific mouse operation. The AI virtual

mouse system was created using the Python programming language, as well as OpenCV, a computer vision library. The model in the proposed AI virtual mouse system makes use of the MediaPipe package for tracking the hands and the tip of the hands, as well as the Autopy, and PyAuto GUI packages for moving around the computer's window screen and performing functions like left click, right click, and scrolling. The proposed model's results demonstrated a very high level of accuracy, and the proposed model can function extremely well in realworld applications using only a CPU and no GPU.

1.1 OBJECTIVE:

The basic objective is to develop a virtual mouse and keyboard using the concepts of hand gesture recognition and image processing which will ultimately move the mouse pointer according to the hand gestures, similarly with the help of the gesture can use keyboard functions which will be defined as per the convenience of the user. Reducing the cost of hardware.

1.2 PROBLEM STATEMENT:

Usage of physical mouse and keyboard will get wear and tear someday, which might be a materialistic damage or a functioning problem.

1.3 EXISTING SYSTEM:

- A physical mouse and keyboard is required to work with the system.
- Mouse is a physical device where it can wear and tear.
- Need a flat surface to use the device effectively.

1.3.1 DRAWBACKS:

- They need a flat surface close to the system.
- If the battery wears out in a wireless mouse, it cannot be used until it has been replaced.

1.4 PROPOSED SYSTEM:

- The system can able to detect the hand gestures.

- Each and every gesture mapped to a mouse functional command.
- We can perform several mouse functions like single click, double click, scroll, drag and drop.
- The Mouse uses a convex hull process for its working, defects are captured or read, using this defects the functions of the mouse are mapped.

1.4.1 ADVANTAGES:

- Reduce hardware cost by eliminating use of physical mouse and keyboard.
- The main advantage of using hand gestures is to interact with computer as a non-contact human computer.
- We can create more mouse functions as we can create many number of gestures easily.
- Adding new functionalities takes less amount of time.

SYSTEM ANALYSIS

Chapter-II

SYSTEM ANALYSIS

2.1 LITERATURE SURVEY:

- S.S. Rautray, A. Agrawal, “Real Time Gesture Recognition System for Interaction in Dynamic Environment,” Vol. 4, pp 595- 599. Human Computer Interaction techniques have become a bottleneck in the effective utilization of the available information flow. The development of user interfaces influences the changes in the Human-Computer Interaction (HCI) - 2019.
- Z. Shujun, “A Hand Gesture Recognition Method Based on Multi- Feature Fusion and Template Matching. Hand gesture recognition system received great attention in the recent few years because of its manifoldness applications and the ability to interact with machine efficiently through human computer interaction - 2012.
- R. Zaman, K. Noor, A. Ibraheem, Hand Gesture Recognition : A Literature Review Gestures can be static (posture or certain pose) which require less computational complexity or dynamic (sequence of postures) which are more complex but suitable for real time environments - 2012.
- Mahajan A.R, International Journal on Science and Technology. Image based one time password is also a factor nowadays to enhance the security of One Time Passwords it also includes machine learning algorithms for detection of image OTP’s in 2012.
- Sadhana Rao, ” Many works are done using openCV technology that image processing for path holes detection to overcome accidents that’s a main problem in many parts of the world - 2010.

2.2 SYSTEM SPECIFICATION:

The following are implementation specifications considered for development of the proposed system:

2.2.1 HARDWARE SPECIFICATION:

- Processor : Intel i3
- Processor speed : 1.80 GHz
- Hard Disk : 1 TB
- RAM : 4 GB
- Web Cam : 30 FPS (Frames Per Second)

2.2.2 SOFTWARE SPECIFICATION:

- Operating System : Windows XP
- Coding Language : PYTHON 3.9.6
- IDE : PYTHON IDLE
- Database : MySQL
- Libraries : Open CV, mediapipe

2.3 FEASIBILITY STUDY:

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

- Economical Feasibility
- Technical Feasibility
- Social Feasibility

2.3.1. ECONOMICAL FEASIBILITY:

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

2.3.2. TECHNICAL FEASIBILITY:

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.3.3 SOCIAL FEASIBILITY:

This aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it.

SYSTEM DESIGN

Chapter – III

SYSTEM DESIGN

3.1 ARCHITECTURE:

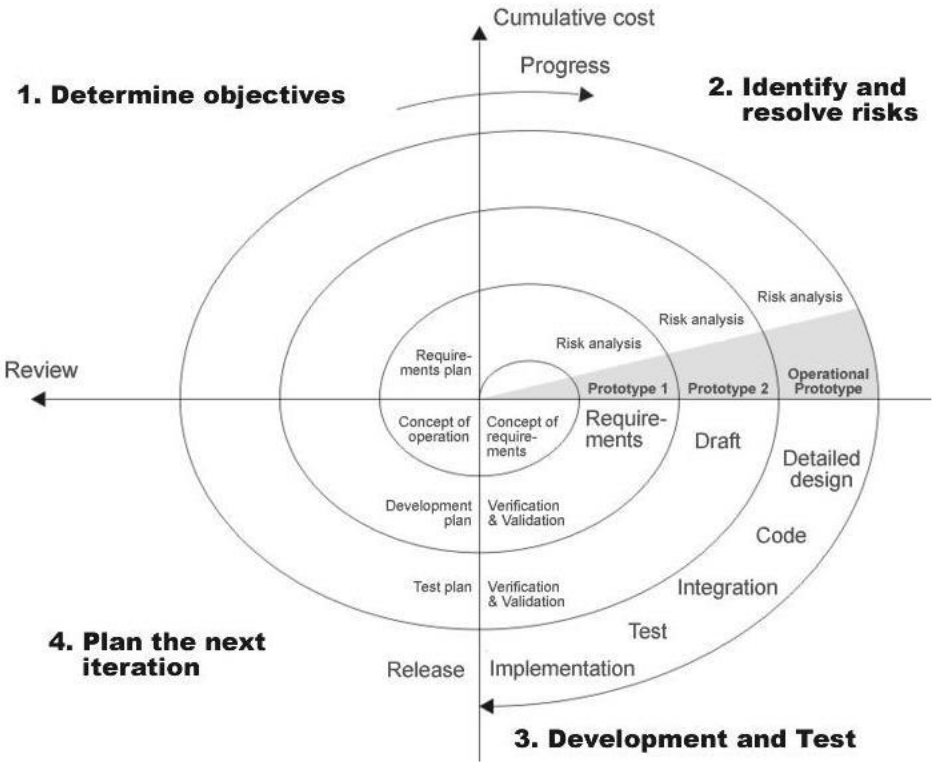


Figure 3.1: Software Development Life Cycle

3.2 MODULES:

The proposed “Trip Advisor” system has 2 modules:

- User
- Admin

- Registration module
- Login module
- Video accessing module

- Gesture recognition module
- Event generation module

3.2.1 REGISTRATION MODULE:

The users must register into the system by entering a username and a password. The username and password is stored in the database.

3.2.2 LOGIN MODULE:

To access the virtual mouse and the keyboard, the users must log in into the system with their credentials. Once the user enters the credentials then the credentials are checked with the data present in the database. If the credentials are valid then only the users have access to the system else the users can not be able to use the virtual mouse and keyboard.

3.2.3 VIDEO ACCESSING MODULE:

By the usage of open CV library methods, we can be able to capture the live video through web cam. We can also perform tasks like face detection, landmark detection and much more.

Open CV is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel. By using this open CV technology we can access the web cam and we can implement several operations on the video frames like adding hue, saturation, contrast and contours etc..,

3.2.4 GESTURE RECOGNITION MODULE:

Media pipe is a library of programming functions developed by google. Google provided many number of models to make several operations like face detection, hand gesture recognition, pose recognition and object detection etc..,

By using the hand gesture recognition models, we can able to detect the gestures and can able to visualize the skeleton view of the hands.

By using the hand gesture recognition models, we can able to detect the gestures of the hands.

3.2.5 EVENT GENERATION MODULE:

Once the gestures are detected, each gesture is mapped to a mouse function. By detecting the gesture, corresponding mouse and keyboard functions can be performed.

We have several mouse functions like single click, double click, drag and drop and scroll. Similarly, each key in the keyboard has its unique functionality.

Based on the gesture, we can perform several mouse and keyboard functionalities by generating the events of mouse and keyboard.

3.3 DESIGN REPRESENTATION:

A UML diagram is a diagram based on the UML (Unified Modeling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system.

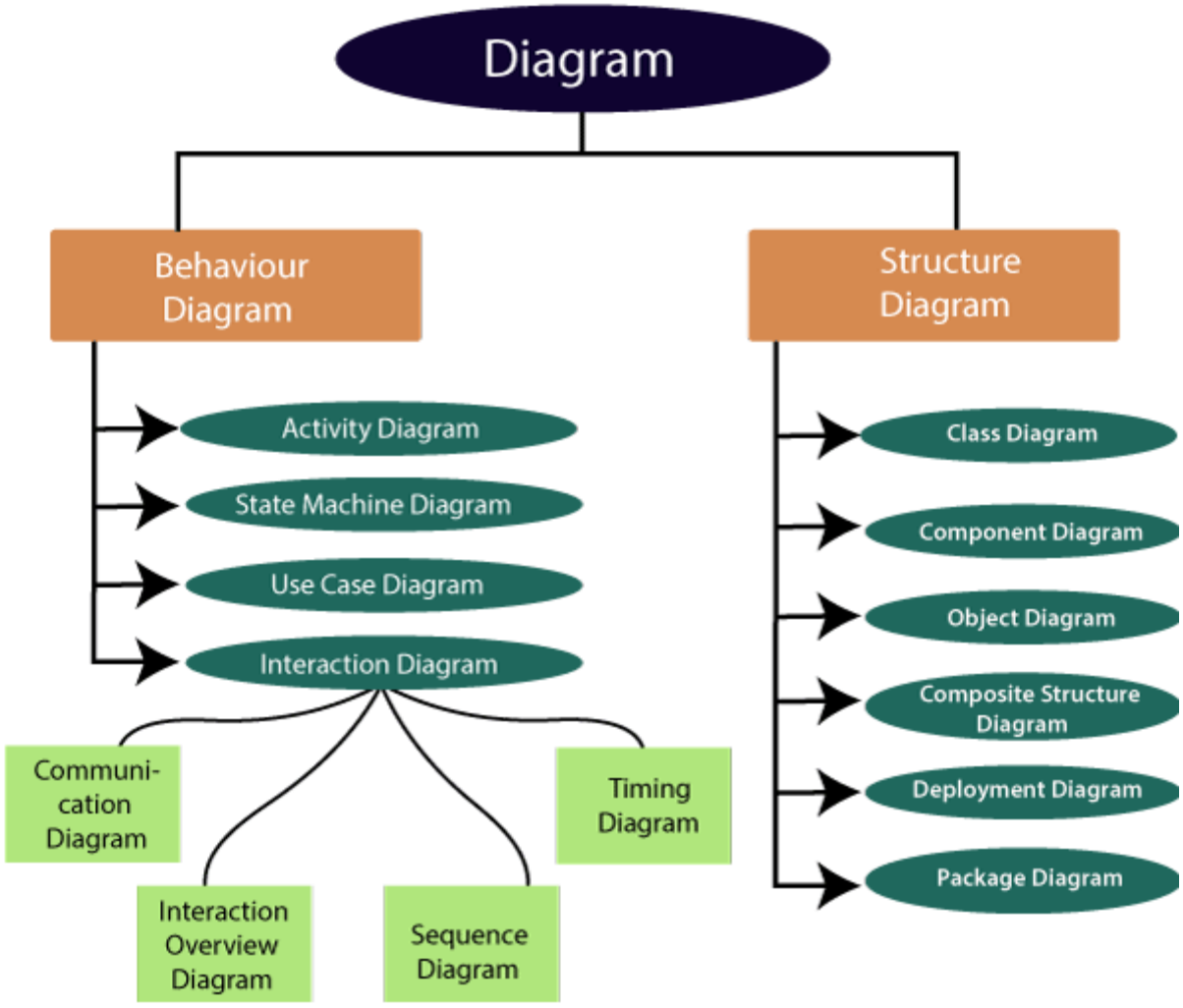


Figure 3.1 UML Architecture

3.3.1 USE CASE DIAGRAM:

It represents the functionality of a system by utilizing actors and use cases. It encapsulates the functional requirement of a system and its association with actors. It portrays the use case view of a system.

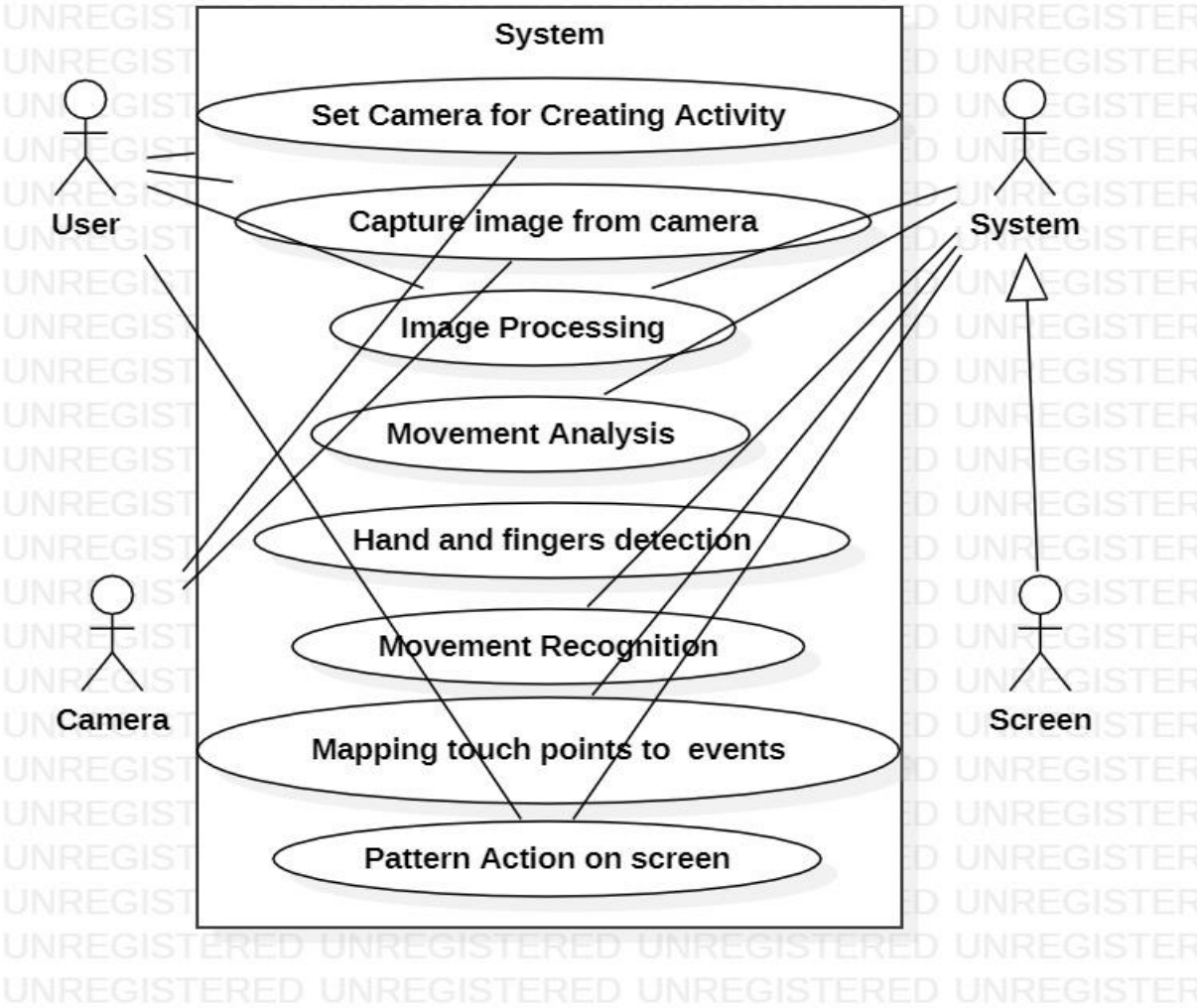


Figure 3.2: Use Case Diagram

3.3.2 CLASS DIAGRAM:

Class diagrams are one of the most widely used diagrams. It is the backbone of all the object-oriented software systems. It depicts the static structure of the system. It displays the system's class, attributes, and methods. It is helpful in recognizing the relation between different objects as well as classes.

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

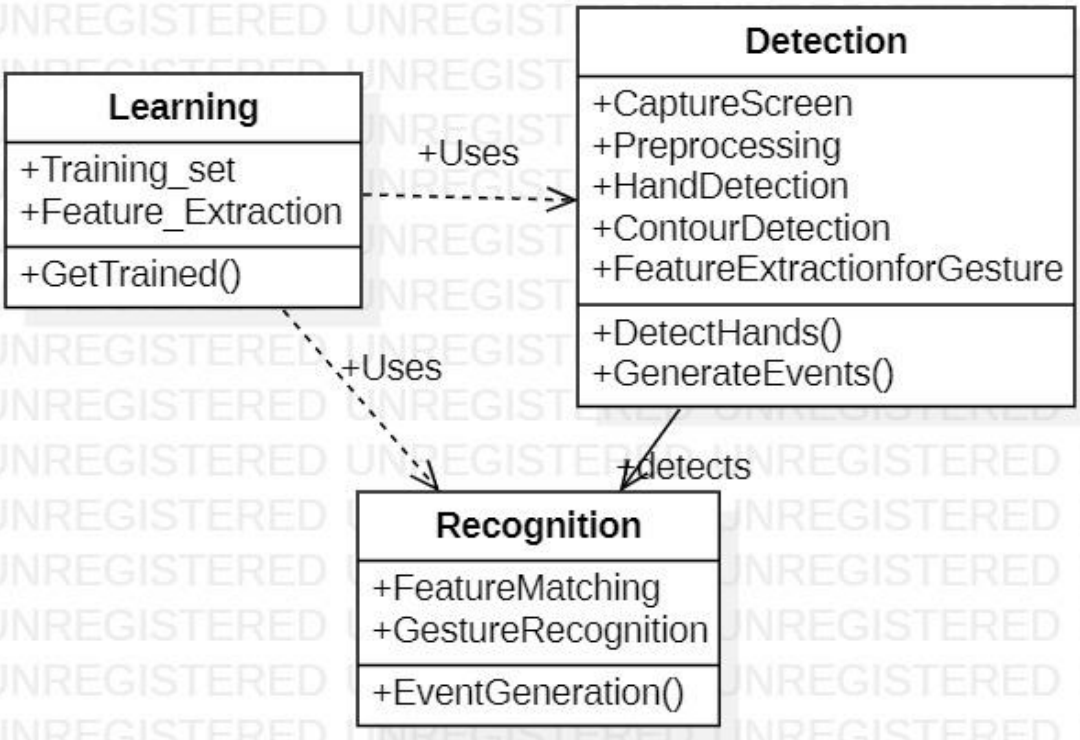


Figure 3.3: Class Diagram

3.3.3 SEQUENCE DIAGRAM:

It shows the interactions between the objects in terms of messages exchanged over time. It delineates in what order and how the object functions are in a system.

A sequence diagram or system sequence diagram (SSD) shows process interactions arranged in time sequence in the field of software engineering. It depicts the processes involved and the sequence of messages exchanged between the processes needed to carry out the functionality. The sequence diagram is a good diagram to use to document a system's requirements and to flush out a system's design. The reason the sequence diagram is so useful is because it shows the interaction logic between the objects in the system in the time order that the interactions take place.

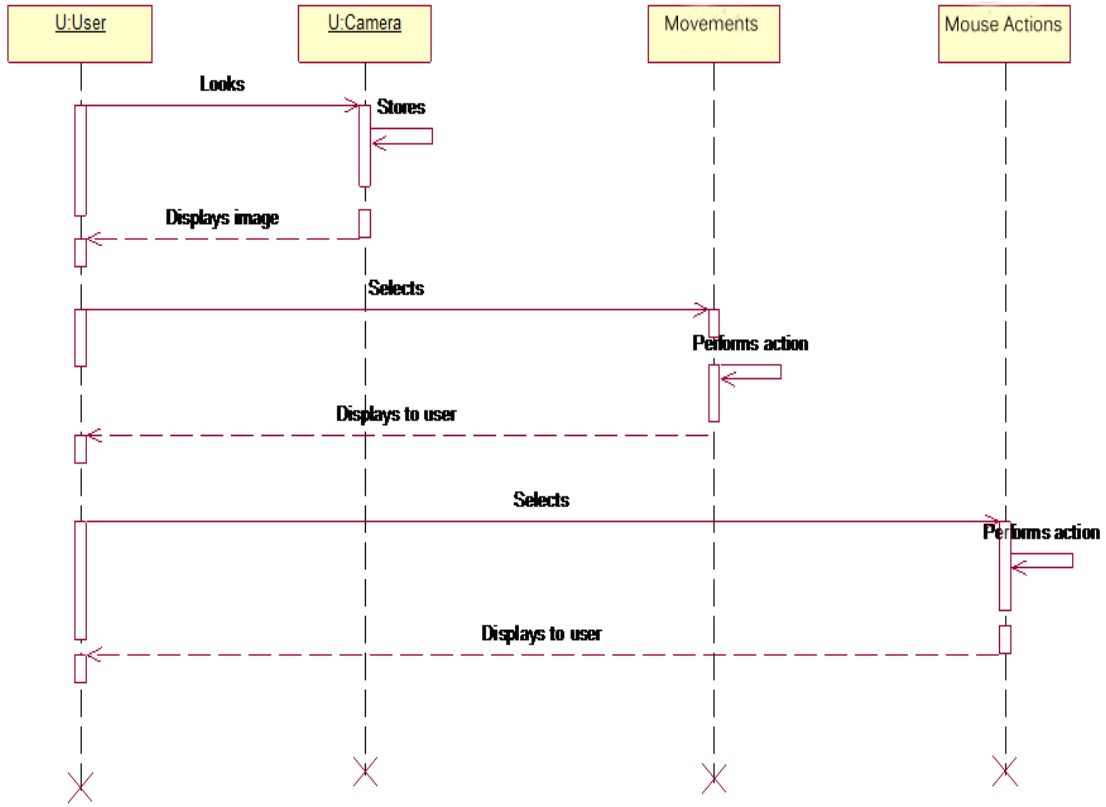


Figure 3.4: Sequence Diagram

3.3.4 DATA FLOW DIAGRAM:

A data flow diagram (DFD) is a **visual representation of the information flow through a process or system**. DFDs help you better understand process or system operation to discover potential problems, improve efficiency, and develop better processes.

A data flow diagram (DFD) illustrates how data is processed by a system in terms of inputs and outputs. As its name indicates its focus is on the flow of information, where data comes from, where it goes and how it gets stored.

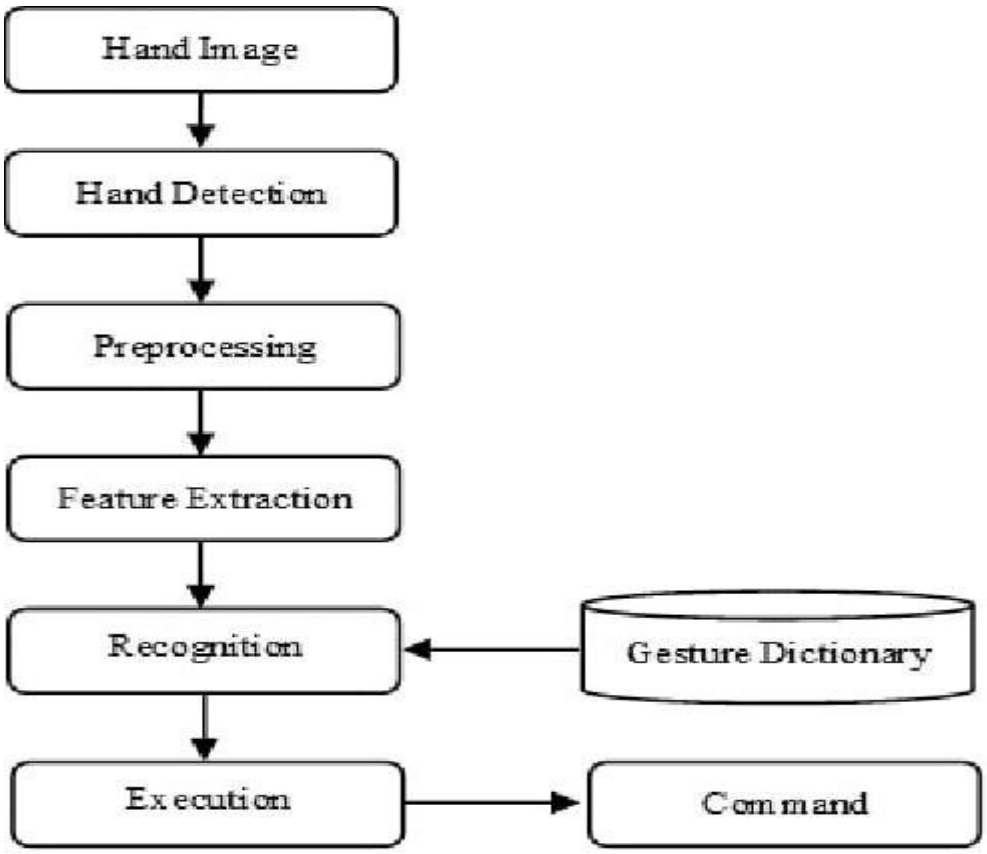


Figure 3.5: Data Flow Diagram

IMPLEMENTATION

Chapter-IV

IMPLEMENTATION

4.1 TECHNOLOGIES:

4.1.1 OPENCV:

Computer vision is a process by which we can understand the images and videos how they are stored and how we can manipulate and retrieve data from them. Computer Vision is the base or mostly used for Artificial Intelligence. Computer-Vision is playing a major role in self-driving cars, robotics as well as in photo correction apps.

OpenCV is the huge open-source library for the computer vision, machine learning, and image processing and 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 handwriting of a human. When it integrated with various libraries, such as NumPy, python is capable of processing the OpenCV array structure for analysis. To Identify image pattern and its various features we use vector space and perform mathematical operations on these features.

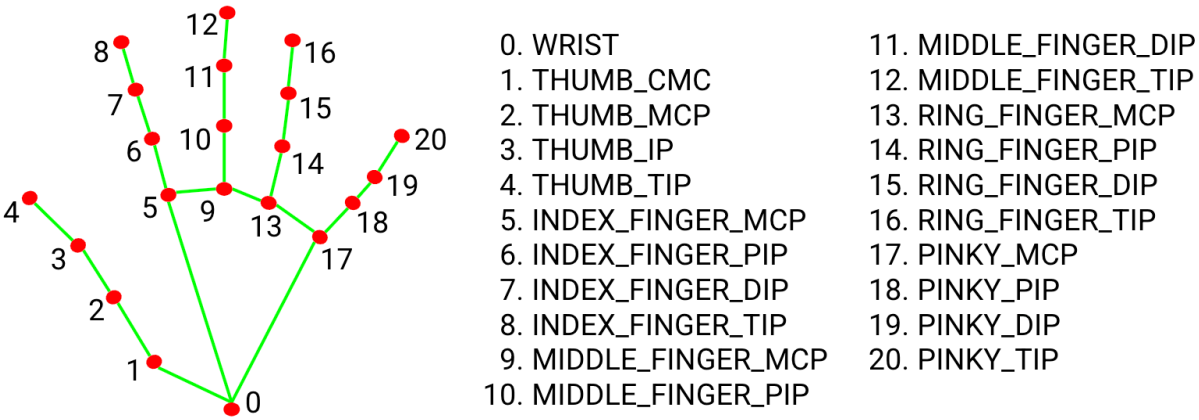


Figure 4.1: Hand Land marks

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.

Applications of OpenCV:

There are lots of applications which are solved using OpenCV, some of them are listed below

- face recognition
- Automated inspection and surveillance
- number of people – count (foot traffic in a mall, etc)
- Vehicle counting on highways along with their speeds
- Interactive art installations
- Anamoly (defect) detection in the manufacturing process (the odd defective products)
- Street view image stitching
- Video/image search and retrieval
- Robot and driver-less car navigation and control
- object recognition
- Medical image analysis
- Movies – 3D structure from motion
- TV Channels advertisement recognition

Open-CV Functionality:

- Image/video I/O, processing, display (core, imgproc, highgui)
- Object/feature detection (objdetect, features2d, nonfree)
- Geometry-based monocular or stereo computer vision (calib3d, stitching, videostab)
- Computational photography (photo, video, superres)
- Machine learning & clustering (ml, flann)
- CUDA acceleration (gpu)

Image-Processing

Image processing is a method to perform some operations on an image, in order to get an enhanced image and or to extract some useful information from it.

If we talk about the basic definition of image processing then **“Image processing is**

the analysis and manipulation of a digitized image, especially in order to improve its quality".

Digital-Image :

An image may be defined as a two-dimensional function f(x, y), where x and y are spatial (plane) coordinates, and the amplitude of fat any pair of coordinates (x, y) is called the intensity or grey level of the image at that point. In another word An image is nothing more than a two-dimensional matrix (3-D in case of coloured images) which is defined by the mathematical function f(x, y) at any point is giving the pixel value at that point of an image, the pixel value describes how bright that pixel is, and what colour it should be. Image processing is basically signal processing in which input is an image and output is image or characteristics according to requirement associated with that image.

Image processing basically includes the following three steps:

1. Importing the image.
2. Analysing and manipulating the image.
3. Output in which result can be altered image or report that is based on image analysis.

4.1.2 PYTHON:

Python is a popular programming language. It was created by Guido van Rossum, and released in 1991.

It is used for:

- web development (server-side),
- software development,
- mathematics.
- system scripting.

What can Python do?

- Python can be used on a server to create web applications.
- Python can be used alongside software to create workflows.
- Python can connect to database systems. It can also read and modify files.
- Python can be used to handle big data and perform complex mathematics.
- Python can be used for rapid prototyping, or for production-ready software development.

Why Python?

- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
- Python has a simple syntax similar to the English language.
- Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
- Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
- Python can be treated in a procedural way, an object-oriented way or a functional way.

Good to know

- The most recent major version of Python is Python 3, which we shall be using in this tutorial. However, Python 2, although not being updated with anything other than security updates, is still quite popular.
- In this tutorial Python will be written in a text editor. It is possible to write Python in an Integrated Development Environment, such as Thonny, Pycharm, Netbeans or Eclipse which are particularly useful when managing larger collections of Python files.

Python Syntax compared to other programming languages

- Python was designed for readability, and has some similarities to the English language with influence from mathematics.
- Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.

- Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

4.1.3 PYAUTOGUI:

Automation plays a huge role in our lives, and it allows us to do one more task during the process. Python provides many useful and advanced libraries that make a Python programmer's life easier. The pyautogui library is one of the extensive collections of the useful methods. In this tutorial, we will learn about the pyautogui library and implement into the code by using its features. So, without further delay, let's briefly introduce the pyautogui Library.

What is the pyautogui Library?

Python 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.

4.1.4 AUTOPY:

AutoPy is a simple, cross-platform GUI automation library for Python. It includes functions for controlling the keyboard and mouse, finding colors and bitmaps on-screen, and displaying alerts.

4.1.5 NUMPY:

NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, fourier transform, and matrices. NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely. NumPy stands for Numerical Python.

Why Use NumPy?

In Python we have lists that serve the purpose of arrays, but they are slow to process. NumPy aims to provide an array object that is up to 50x faster than traditional Python lists. The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy. Arrays are very frequently used in data science, where speed and resources are very important.

Why is NumPy Faster Than Lists?

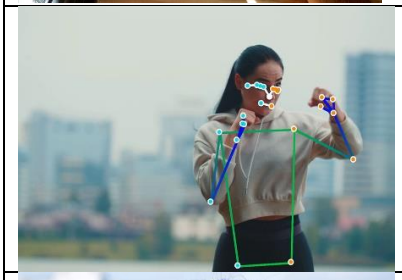
NumPy arrays are stored at one continuous place in memory unlike lists, so processes can access and manipulate them very efficiently. This behavior is called locality of reference in computer science. This is the main reason why NumPy is faster than lists. Also it is optimized to work with latest CPU architectures.

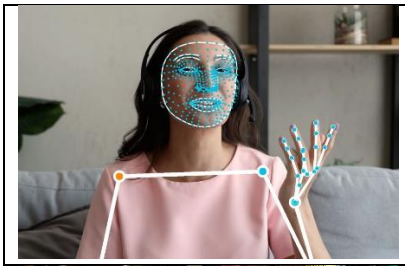
Which Language is NumPy written in?

NumPy is a Python library and is written partially in Python, but most of the parts that require fast computation are written in C or C++.

4.1.6 MEDIAPIPE:

MediaPipe offers open source cross-platform, customizable ML solutions for live and streaming media.

	**Selfie Segmentation** Provides segmentation masks for prominent humans in the scene
	**Face Mesh** 468 face landmarks in 3D with multi-face support
	**Hand Tracking** 21 landmarks in 3D with multi-hand support, based on high-performance palm detection and hand landmark model
	**Human Pose Detection and Tracking** High-fidelity human body pose tracking, inferring up to 33 3D full-body landmarks from RGB video frames
	**Hair Segmentation** Super realistic real-time hair recoloring
	**Object Detection and Tracking** Detection and tracking of objects in video in a single pipeline
	**Face Detection** Ultra lightweight face detector with 6 landmarks and multi-face support

	**Holistic Tracking** Simultaneous and semantically consistent tracking of 33 pose, 21 per-hand, and 468 facial landmarks
	**3D Object Detection** Detection and 3D pose estimation of everyday objects like shoes and chairs

4.1.7 TKINTER:

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is the most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter is the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.

4.1.8 MATH:

Python has a built-in module that you can use for mathematical tasks. The math module has a set of methods and constants.

4.2 SAMPLE CODE:

MOUSE

```
import pyautogui                # mouse movements
import numpy as np              # Data storage - array format
import HandTracking as ht       # module - Google ( mediapipe )
import cv2                      # open source computer vision - intel
import time                     # per second - photos capture
import autopy                   # mouse movements
pTime = 0                       # Used to calculate frame rate
wid = 640                       # Width of Camera
```

```
hei = 480                       # Height of Camera
frameR = 100                    # Frame Rate
smoothening = 8                 # Smoothening Factor
prev_x, prev_y = 0, 0           # Previous coordinates
curr_x, curr_y = 0, 0           # Current coordinates
cap = cv2.VideoCapture(0)                   # Getting video feed from the webcam
cap.set(3, wid)                             # Adjusting size
cap.set(4, hei)
detector = ht.handDetector(maxHands=2)              # Detecting one hand at max
screen_width, screen_height = autopy.screen.size()  # Getting the screen size
while True:
    success, img = cap.read()
    img = detector.findHands(img)                   # Finding the hand
    lmlist, bbox = detector.findPosition(img)       # Getting position of hand
    if len(lmlist) != 0:
        x1, y1 = lmlist[8][1:]
        x2, y2 = lmlist[12][1:]
        fingers = detector.fingersUp()              # Checking if fingers are upwards
        cv2.rectangle(img, (frameR, frameR), (wid - frameR, hei - frameR), (255, 0, 255),2)
if fingers == [0, 1, 0, 0, 0]:
            x3 = np.interp(x1, (frameR, wid - frameR), (0, screen_width))
            y3 = np.interp(y1, (frameR, hei - frameR), (0, screen_height))
            curr_x = prev_x + (x3 - prev_x) / smoothening
            curr_y = prev_y + (y3 - prev_y) / smoothening
            autopy.mouse.move(screen_width - curr_x, curr_y)        # Moving the cursor
            cv2.circle(img, (x1, y1), 7, (255, 0, 255), cv2.FILLED)
            prev_x, prev_y = curr_x, curr_y
        # if fingers[1] == 1 and fingers[2] == 1 and fingers[3] == 0:
        if fingers == [0, 1, 1, 0, 0]:
            x3 = np.interp(x1, (frameR, wid - frameR), (0, screen_width))
            y3 = np.interp(y1, (frameR, hei - frameR), (0, screen_height))
            curr_x = prev_x + (x3 - prev_x) / smoothening
            curr_y = prev_y + (y3 - prev_y) / smoothening
            autopy.mouse.move(screen_width - curr_x, curr_y)        # Moving the cursor
```

```
            cv2.circle(img, (x1, y1), 7, (255, 0, 255), cv2.FILLED)
            prev_x, prev_y = curr_x, curr_y
            length, img, lineInfo = detector.findDistance(8, 12, img)
            if length < 40
                cv2.circle(img, (lineInfo[4], lineInfo[5]), 15, (0, 255, 0), cv2.FILLED)
                autopy.mouse.click()
        if fingers == [0, 1, 1, 1, 1]:  pyautogui.mouseDown(button='right')
        if fingers == [0, 1, 1, 1, 0]: pyautogui.mouseDown(button='left')
        if fingers == [0, 0, 0, 0, 1]:  pyautogui.scroll(10, _pause=False)
        if fingers == [1, 0, 0, 0, 0]: pyautogui.scroll(-10, _pause=True)
         if fingers == [0, 0, 0, 0, 0]:  break
    cTime = time.time()
    fps = 1 / (cTime - pTime)
    pTime = cTime
    cv2.putText(img, str(int(fps)), (20, 50), cv2.FONT_HERSHEY_PLAIN, 3, (255, 0, 0), 3)
    cv2.imshow("Video Output", img)
    cv2.waitKey(1)
```

KEYBOARD

```
pTime = 0
wid = 1280
hei = 960
frameR = 100
smoothening = 8
prev_x, prev_y = 0, 0
curr_x, curr_y = 0, 0
cap = cv2.VideoCapture(0)
cap.set(3, wid)
cap.set(4, hei)
inp = ""
detector = ht.handDetector(maxHands=2)
screen_width, screen_height = autopy.screen.size()
coord = []
```

```
kb = [list(map(str, '1 2 3 4 5 6 7 8 9 0'.split(' '))), list(map(str, 'Q W E R T Y U I O P '.split(' '))), list(map(str, 'A S D F G H J K L ;'.split(' '))),
     list(map(str, 'Z X C V B N M < > <-'.split(' '))), list(map(str, '! @ # $ SPACE ; & * ( CLS'.split(' ')))]
kby = [380, 413, 449, 483, 518, 551]
while True:
    success, img = cap.read()
    img = detector.findHands(img)
    cv2.namedWindow("Video Output")
    lmlist, bbox = detector.findPosition(img)
    cv2.moveWindow("Video Output", 0, 0)
    cv2.putText(img, inp, (60, 500), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0), 3)
    # pyautogui.write(inp)
    if cv2.waitKey(1) & 0xFF == ord('w'):
        print(f'({pyautogui.position().x-300},{pyautogui.position().y-190})')

    if len(lmlist) != 0:
        x1, y1 = lmlist[8][1:]
        x2, y2 = lmlist[12][1:]
        fingers = detector.fingersUp()
        cv2.rectangle(img, (frameR, frameR), (wid - frameR, hei - frameR), (255, 0, 255),2)
        if fingers[1] == 1 and fingers[2] == 0 and fingers[3] == 0:
            x3 = np.interp(x1, (frameR, wid - frameR), (0, screen_width))
            y3 = np.interp(y1, (frameR, hei - frameR), (0, screen_height))
            curr_x = prev_x + (x3 - prev_x) / smoothening
            curr_y = prev_y + (y3 - prev_y) / smoothening
            autopy.mouse.move(screen_width - curr_x, curr_y)
            cv2.circle(img, (x1, y1), 7, (255, 0, 255), cv2.FILLED)
            prev_x, prev_y = curr_x, curr_y
        if fingers[1] == 1 and fingers[2] == 1 and fingers[3] == 0:
            x3 = np.interp(x1, (frameR, wid - frameR), (0, screen_width))
            y3 = np.interp(y1, (frameR, hei - frameR), (0, screen_height))
            curr_x = prev_x + (x3 - prev_x) / smoothening
            curr_y = prev_y + (y3 - prev_y) / smoothening
```

```
        autopy.mouse.move(screen_width - curr_x, curr_y)
        cv2.circle(img, (x1, y1), 7, (255, 0, 255), cv2.FILLED)
        prev_x, prev_y = curr_x, curr_y
        length, img, lineInfo = detector.findDistance(8, 12, img)
        if length < 40:
            cv2.circle(img, (lineInfo[4], lineInfo[5]), 15, (0, 255, 0), cv2.FILLED)
            autopy.mouse.click()
            tkb = 0
            for kbxy in range(5):
                if pyautogui.position().x-300 >= -278 and pyautogui.position().x-300 <= -154
and pyautogui.position().y-190 >= kby[kbxy] and pyautogui.position().y-190 <=
kby[kbxy+1]:
                    print(kb[kbxy][0])
                    inp += kb[kbxy][0]
                elif pyautogui.position().x-300 >= -154 and pyautogui.position().x-300 <= -29
and pyautogui.position().y-190 >= kby[kbxy] and pyautogui.position().y-190 <=
kby[kbxy+1]:
                    print(kb[kbxy][1])
                    inp += kb[kbxy][1]
                elif pyautogui.position().x-300 >= -29 and pyautogui.position().x-300 <= 97 and
pyautogui.position().y-190 >= kby[kbxy] and pyautogui.position().y-190 <= kby[kbxy+1]:
                    print(kb[kbxy][2])
                    inp += kb[kbxy][2]
                elif pyautogui.position().x-300 >= 97 and pyautogui.position().x-300 <= 221 and
pyautogui.position().y-190 >= kby[kbxy] and pyautogui.position().y-190 <= kby[kbxy+1]:
                    print(kb[kbxy][3])
                    inp += kb[kbxy][3]
                elif pyautogui.position().x-300 >= 221 and pyautogui.position().x-300 <= 346
and pyautogui.position().y-190 >= kby[kbxy] and pyautogui.position().y-190 <=
kby[kbxy+1]:
                    if kbxy == 4: inp += " "
                    else:
                        print(kb[kbxy][4])
```

```
                inp += kb[kbxy][4]
            elif pyautogui.position().x-300 >= 346 and pyautogui.position().x-300 <= 472
and pyautogui.position().y-190 >= kby[kbxy] and pyautogui.position().y-190 <=
kby[kbxy+1]:
                if kbxy == 4: inp += " "
                else:
                    print(kb[kbxy][5])
                    inp += kb[kbxy][5]
            elif pyautogui.position().x-300 >= 472 and pyautogui.position().x-300 <= 596
and pyautogui.position().y-190 >= kby[kbxy] and pyautogui.position().y-190 <=
kby[kbxy+1]:
                print(kb[kbxy][6])
                inp += kb[kbxy][6]
            elif pyautogui.position().x-300 >= 596 and pyautogui.position().x-300 <= 720
and pyautogui.position().y-190 >= kby[kbxy] and pyautogui.position().y-190 <=
kby[kbxy+1]:
                print(kb[kbxy][7])
                inp += kb[kbxy][7]
            elif pyautogui.position().x-300 >= 720 and pyautogui.position().x-300 <= 847
and pyautogui.position().y-190 >= kby[kbxy] and pyautogui.position().y-190 <=
kby[kbxy+1]:
                print(kb[kbxy][8])
                inp += kb[kbxy][8]
            elif pyautogui.position().x-300 >= 847 and pyautogui.position().x-300 <= 968
and pyautogui.position().y-190 >= kby[kbxy] and pyautogui.position().y-190 <=
kby[kbxy+1]:
                if kbxy == 3: inp = inp[:-1]
                elif kbxy == 4: inp = ""
                else:
                    print(kb[kbxy][9])
                    inp += kb[kbxy][9]
        if fingers == [0, 1, 1, 1, 1]:
            pyautogui.mouseDown(button='right')
        if fingers == [0, 1, 1, 1, 0]:
```

```
            pyautogui.mouseDown(button='left')
        if fingers == [0, 0, 0, 0, 1]:
            pyautogui.scroll(10, _pause=False)
        if fingers == [1, 0, 0, 0, 0]:
            pyautogui.scroll(-10, _pause=True)
    cv2.rectangle(img, (10, 525), (1260, 700), (100, 100, 100), -1)
    cv2.line(img, (10, 525), (1260, 525), (255, 255, 255), 1)
    cv2.line(img, (10, 560), (1260, 560), (255, 255, 255), 1)
    cv2.line(img, (10, 595), (1260, 595), (255, 255, 255), 1)
    cv2.line(img, (10, 630), (1260, 630), (255, 255, 255), 1)
    cv2.line(img, (10, 665), (1260, 665), (255, 255, 255), 1)
    cv2.line(img, (10, 700), (1260, 700), (255, 255, 255), 1)
    cv2.line(img, (10, 525), (10, 700), (255, 255, 255), 5)
    cv2.line(img, (135, 525), (135, 700), (255, 255, 255), 5)
    cv2.line(img, (260, 525), (260, 700), (255, 255, 255), 5)
    cv2.line(img, (385, 525), (385, 700), (255, 255, 255), 5)
    cv2.line(img, (510, 525), (510, 700), (255, 255, 255), 5)
    cv2.line(img, (635, 525), (635, 665), (255, 255, 255), 5) #700
    cv2.line(img, (760, 525), (760, 700), (255, 255, 255), 5)
    cv2.line(img, (885, 525), (885, 700), (255, 255, 255), 5)
    cv2.line(img, (1010, 525), (1010, 700), (255, 255, 255), 5)
    cv2.line(img, (1135, 525), (1135, 700), (255, 255, 255), 5)
    cv2.line(img, (1260, 525), (1260, 700), (255, 255, 255), 5)
    ky, thickness = 555, 3
    for i in range(5):
        cv2.putText(img, kb[i][0], (60, ky), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0,
0),thickness)
        cv2.putText(img, kb[i][1], (180, ky), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0),
thickness)
        cv2.putText(img, kb[i][2], (310, ky), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0),
thickness)
        cv2.putText(img, kb[i][3], (440, ky), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0),
thickness)
        cv2.putText(img, kb[i][4], (570, ky), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0),
```

```
thickness)
        cv2.putText(img, kb[i][5], (690, ky), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0),
thickness)
        cv2.putText(img, kb[i][6], (810, ky), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0),
thickness)
        cv2.putText(img, kb[i][7], (940, ky), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0),
thickness)
        cv2.putText(img, kb[i][8], (1060, ky), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0),
thickness)
        cv2.putText(img, kb[i][9], (1180, ky), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0),
thickness)
        ky += 35
    cTime = time.time()
    fps = 1 / (cTime - pTime)
    pTime = cTime
    cv2.putText(img, str(int(fps)), (20, 50),cv2.FONT_HERSHEY_PLAIN, 3, (255, 0, 0), 3)
    cv2.imshow("Video Output", img)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        print(coord)
        break
cap.release()
cv2.destroyAllWindows()
```

TESTING

Chapter-V

TESTING

5.1 METHODOLOGIES

Testing is the process of executing a program to find errors. To make our software perform well it should be error-free. If testing is done successfully it will remove all the errors from the software.

Principles of Testing:-

1. All the tests should meet the customer requirements.
2. To make our software testing should be performed by a third party.
3. Exhaustive testing is not possible. As we need the optimal amount of testing based on the risk assessment of the application.
4. All the tests to be conducted should be planned before implementing it.
5. It follows the Pareto rule (80/20 rule) which states that 80% of errors come from 20% of program components.
6. Start testing with small parts and extend it to large parts.

5.1.1 UNIT TESTING:

Unit testing involves the testing of each unit or an individual component of the software application. It is the first level of functional testing. The aim behind unit testing is to validate unit components with its performance.

A unit is a single testable part of a software system and tested during the development phase of the application software. The purpose of unit testing is to test the correctness of isolated code. A unit component is an individual function or code of the application.

White box testing approach used for unit testing and usually done by the developers. Whenever the application is ready and given to the Test engineer, he/she will start checking every component of the module or module of the application independently or one by one, and this process is known as **Unit testing** or **components testing**.

5.1.2 INTEGRATION TESTING:

Integration testing is the second level of the software testing process comes after unit testing. In this testing, units or individual components of the software are tested in a group. The focus of the integration testing level is to expose defects at the time of interaction between integrated components or units.

Unit testing uses modules for testing purpose, and these modules are combined and tested in integration testing. The Software is developed with a number of software modules that are coded by different coders or programmers. The goal of integration testing is to check the correctness of communication among all the modules.

Once all the components or modules are working independently, then we need to check the data flow between the dependent modules is known as **integration testing**.

5.1.3 VALIDATION TESTING:

Validation testing is testing where tester performed functional and non-functional testing. Here **functional testing** includes Unit Testing (UT), Integration Testing (IT) and System Testing (ST), and **non-functional** testing includes User acceptance testing (UAT).

Validation testing is also known as dynamic testing, where we are ensuring that **"we have developed the product right."** And it also checks that the software meets the business needs of the client.

5.1.4 SYSTEM TESTING:

System Testing includes testing of a fully integrated software system. Generally, a computer system is made with the integration of software (any software is only a single element of a computer system). The software is developed in units and then interfaced with other software and hardware to create a complete computer system. In other words, a computer system consists of a group of software to perform the various tasks, but only software cannot perform the task; for that software must be interfaced with compatible hardware. System testing is a series of different type of tests with the purpose to exercise and examine the full working of an integrated software computer system against requirements.

5.2 TEST CASES:

Inputs	Expected Output	Actual Output	Result
No hands in-front of camera	No detection of hands	No detection of hands	Pass
Single Hand	Detected	Detected	Pass
Two Hands	Detected	Detected	Pass
More Than Two Hands	Only Two Hands Detected	Only Two Hands Detected	Pass
Single Finger Up [0,1,0,0,0]	Mouse moved	Mouse Moved	Pass
Two Fingers Up [0,1,1,0,0]	Mouse movement and Single click	Mouse movement and single click	Pass
Thumb Finger Up [1,0,0,0,0]	Scroll Up	Scroll Up	Pass
Little Finger Up [0,0,0,0,1]	Scroll Down	Scroll Down	Pass
All Fingers Down [0,0,0,0,0]	Application Closed	Application Closed	Pass

Table 5.1: Test Cases

DEPLOYMENT

Chapter-VI

DEPLOYMENT

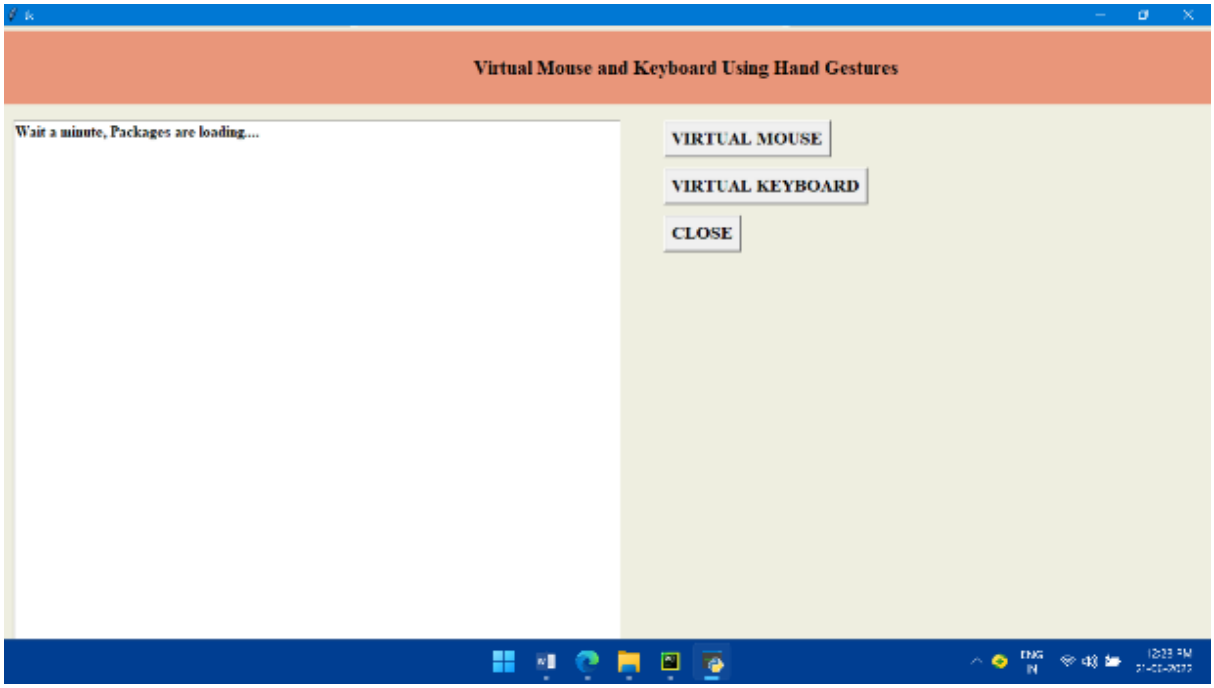


Figure 6.1: Home Page

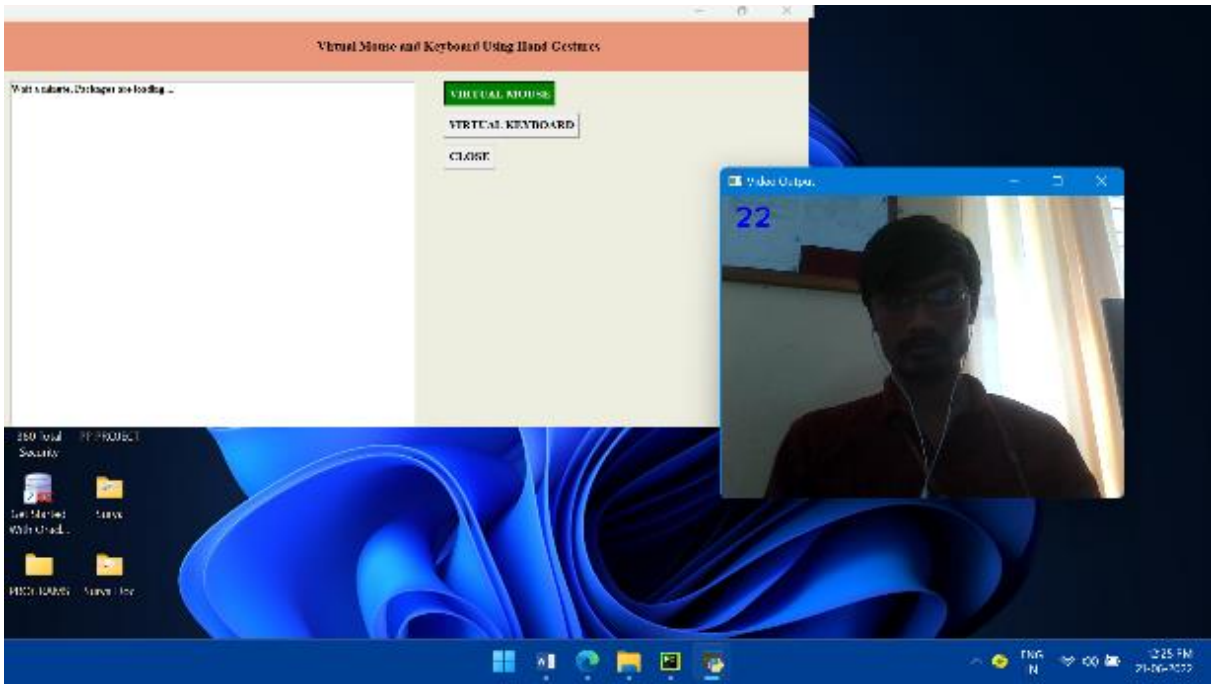


Figure 6.2: Camera Accessed

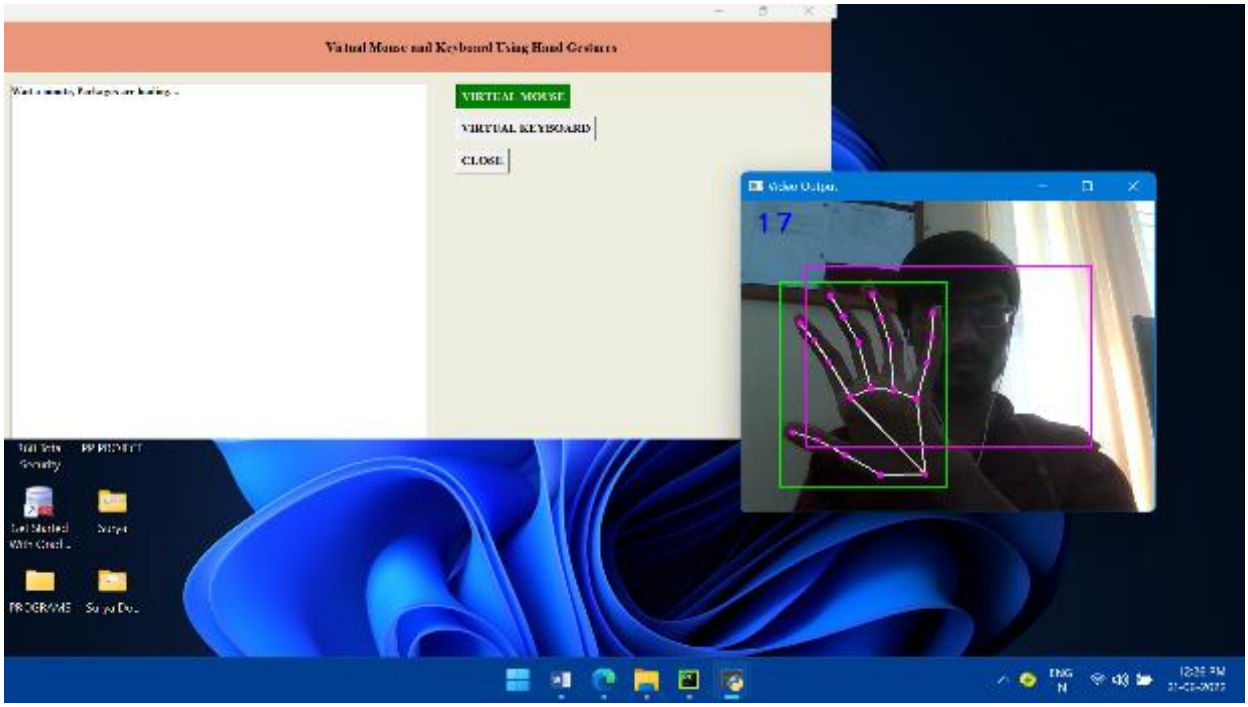


Figure 6.3: Single Hand Detection

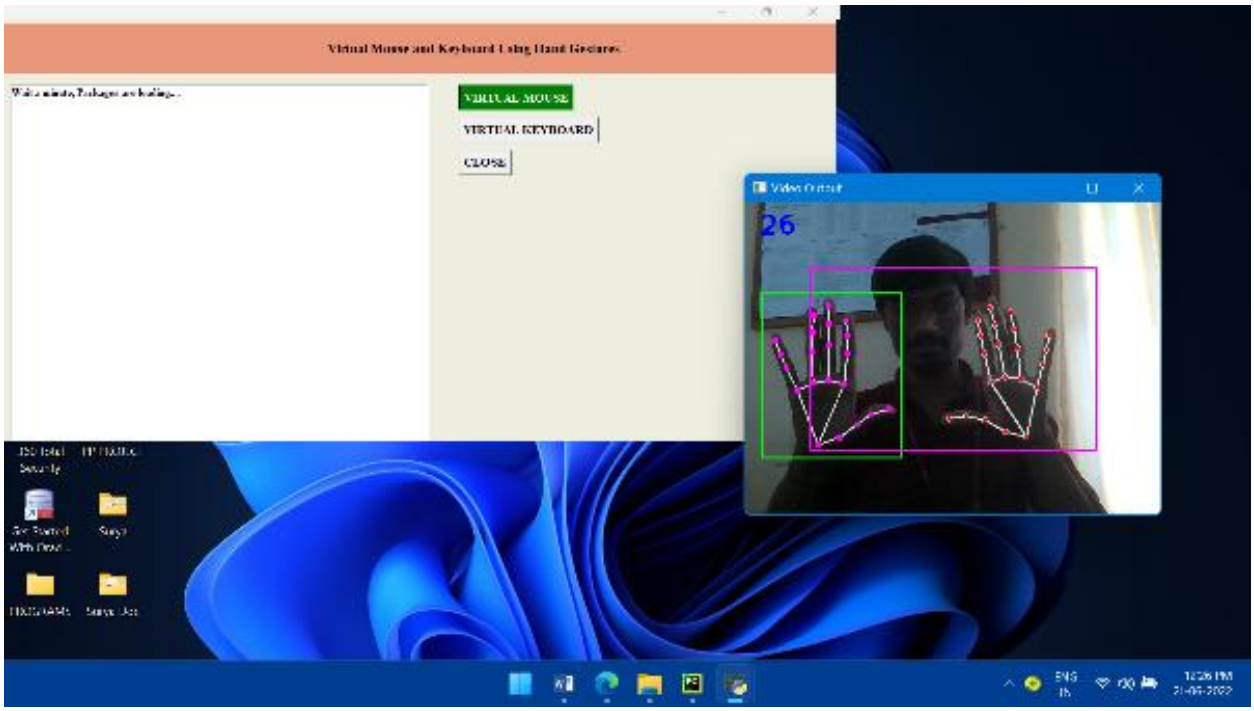


Figure 6.4: Two Hands Detection

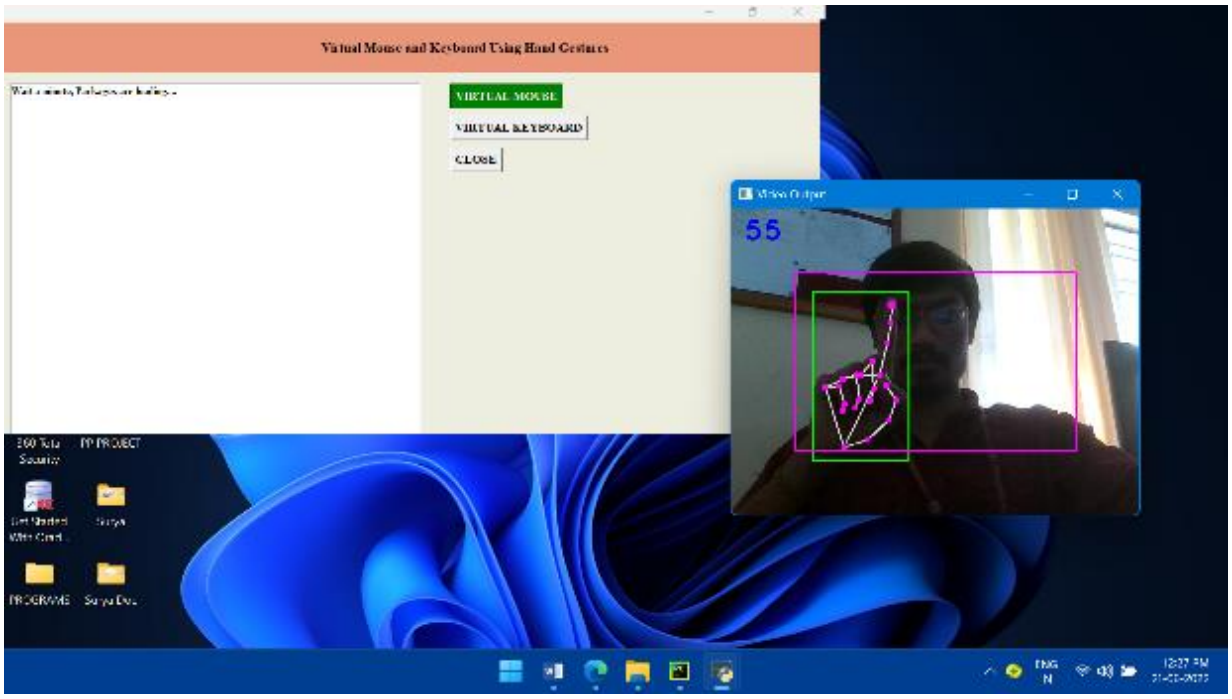


Figure 6.5: Mouse Movement Started

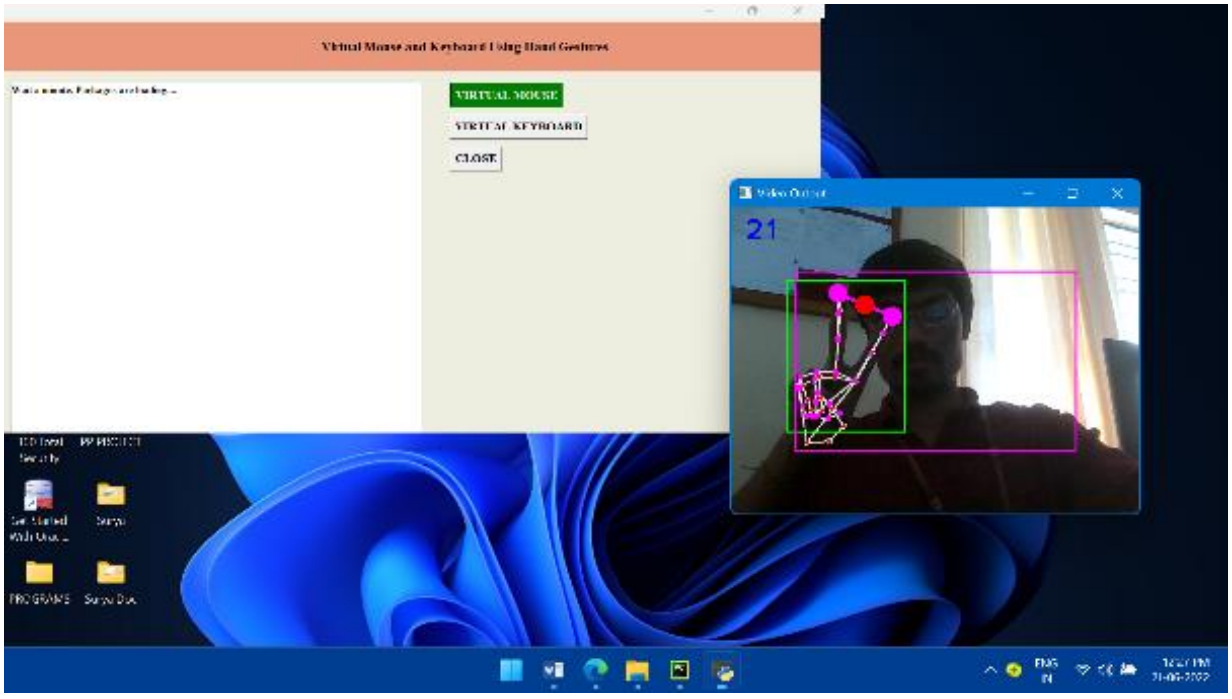


Figure 6.6: Single Click

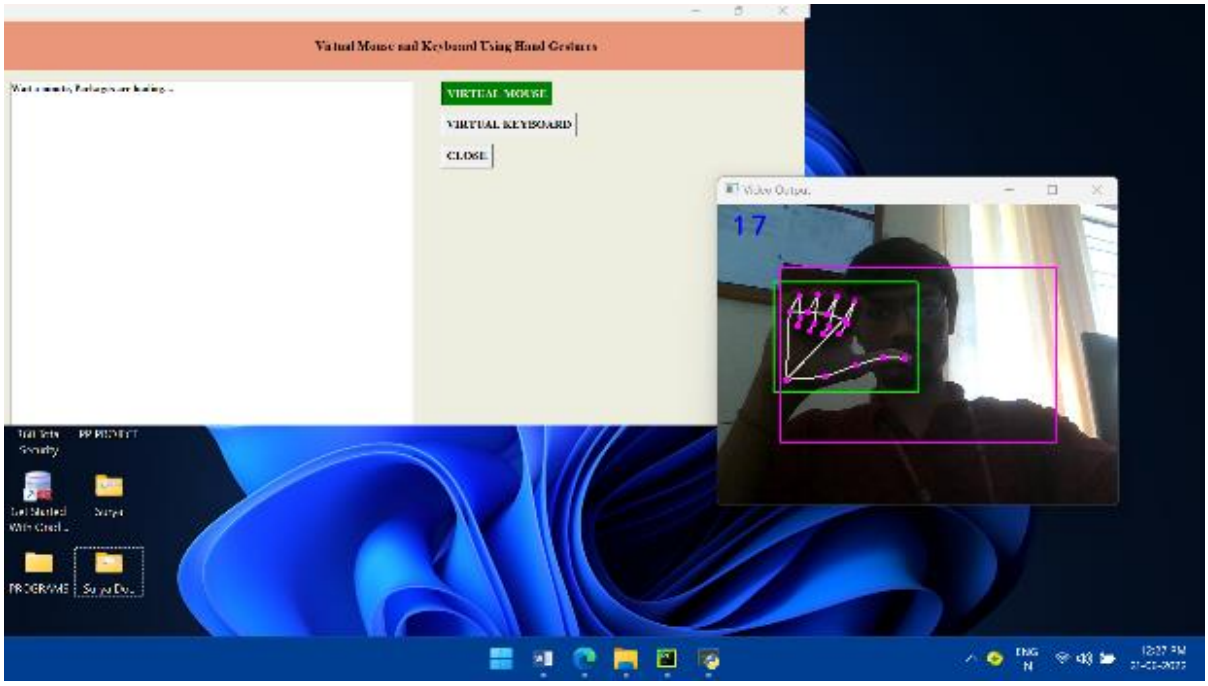


Figure 6.7: Scrolling Up

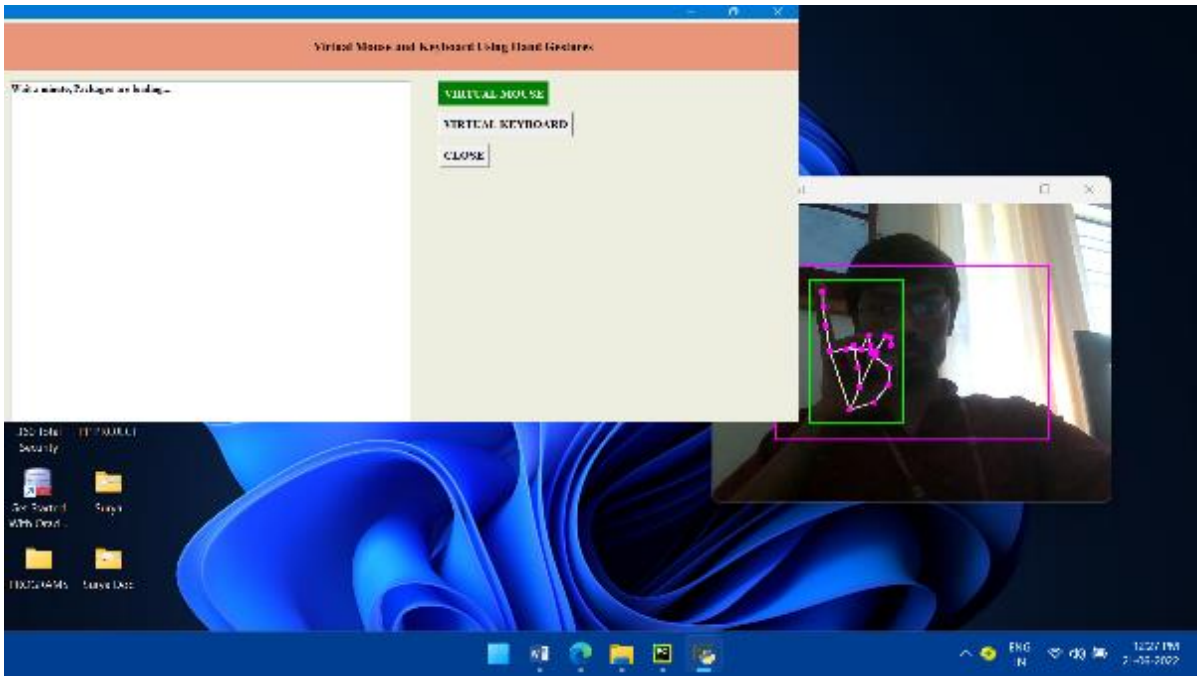


Figure 6.8: Scrolling Down

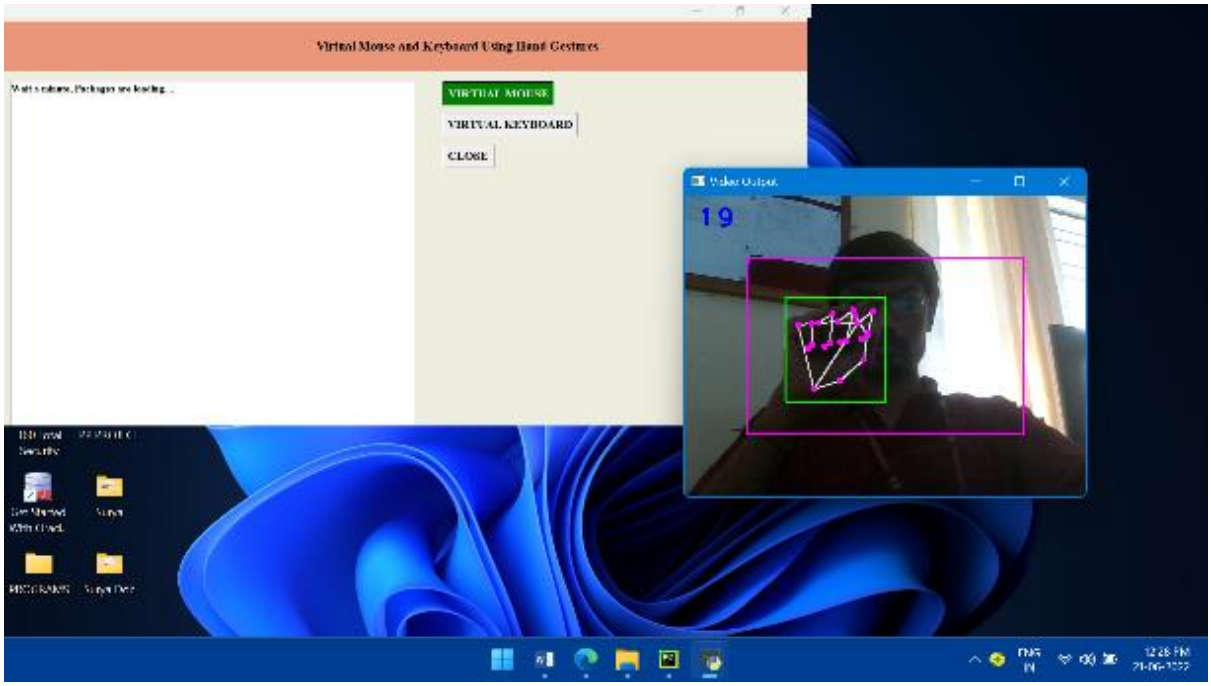


Figure 6.9: Camera Stopped

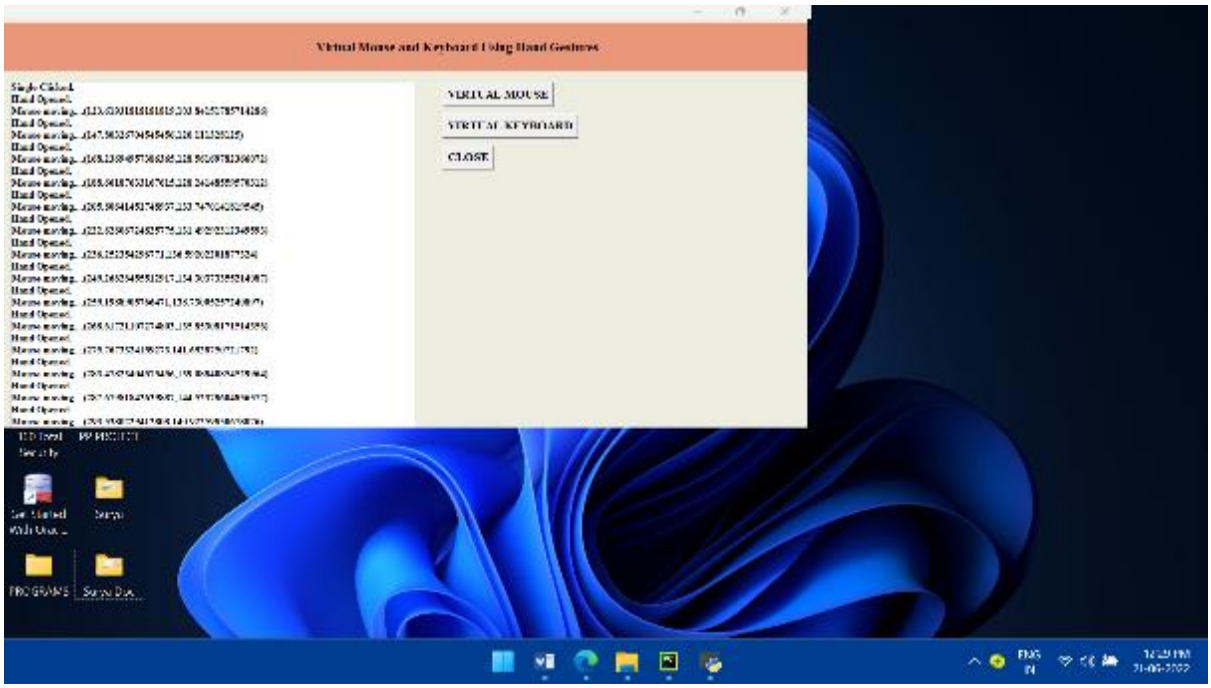


Figure 6.10: Application output

CONCLUSION

Chapter-VII

CONCLUSION

In conclusion, a system that recognizes hand gestures and eliminates the need for a mouse and keyboard. This includes mouse cursor movement, drag and click, and keyboard features such as printing alphabets and other keyboard tasks. To separate the color/image of the hand from its background, the skin segmentation procedure is used. The method of removing an arm efficiently overcomes the problem of capturing the entire body in the camera. In summary, the suggested method can detect and recognize hand gestures in order to control mouse and keyboard functions as well as generate a realistic user interface. 3D printing, architectural renderings, and even medical procedures may all be done from anywhere to anyone. This project is simple to build and has a wide range of applications in medical science where computation is necessary but cannot be fully performed owing to a lack of humancomputer Interface.

FUTURE ENHANCEMENTS

Chapter-VII

FUTURE ENHANCEMENTS

The work is to make PC understanding projects that can deal with continuous issues and to achieve targets of the affiliations and regular daily existence as well as individuals. There is a degree in encouraging the machines games, talk affirmation machine, language revelation, PC vision, ace systems, progressed mechanics, etc. the more you learn about AI sciences, for instance material science or science, the better. For the normal ways of managing Artificial Intelligence, find out about mind science and the tangible framework. Get to know some Machine vernaculars. It is savvy to focus on one crucial machine language. Occupations are commonly to depend after getting the programming vernaculars. Calling decisions in AI where student can land positions at Occupation will be offered like: Game Programmer, Robotic, Scientist, Computer Scientist and data analyst.

BIBLIOGRAPHY

BIBLIOGRAPHY

- **Sugnik roy**, Gesture recognition based virtual mouse and keyboard using convex hull method, 2020.
- **S.S. Rautaray**, A. Agrawal, "Real Time Gesture Recognition System for Interaction in Dynamic Environment," Procedia Technology, Vol. 4, pp 595- 599, 2019.
- **L.Yun, Z. Lifeng, Z. Shujun**, "A Hand Gesture Recognition Method Based on MultiFeature Fusion and Template Matching", Procedia Engineering, Vol. 29, 2012.
- **Mahajan A.R**,"A gestural user interface to Interact with the situation of taking into the whole body into the camera. In computer system", International Journal on Science and Technology - 2012.
- **R. Zaman, K**. Noor, A. Ibraheem, "Hand Gesture Recognition:A Literature Review", International Journal of Artificial Intelligence & Applications (IJAIA), Vol.3, No.4, July 2012.
- **Sadhana Rao**," Sixth Sense Technology", Proceedings of the International Conference on Communication and Computational is utilized to separate the colour/image of hand with its Intelligence– 2010.