VIRTUAL MOUSE USING HAND TRACKING

A Project Work-II Report

Submitted in partial fulfillment of requirement of the

Degree of

BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE & ENGINEERING

BY

Nishita Jain (EN19CS301227)

Under the Guidance of

Dr. Harsh Pratap Singh

Department of Computer Science & Engineering
Faculty of Engineering
MEDI-CAPS UNIVERSITY, INDORE- 453331

JAN-JUNE 2023

Report Approval

The project work “Virtual Mouse Using Hand Tracking” is hereby approved as a creditable study of an engineering subject carried out and presented in a manner satisfactory to warrant its acceptance as a prerequisite for the Degree for which it has been submitted.

It is to be understood that by this approval the undersigned does not endorse or approve any statement made, opinion expressed, or conclusion drawn therein; but approve the “Project Report” only for the purpose for which it has been submitted.

Internal Examiner

Name: Dr. Harsh Pratap Singh

Designation: Assistant Professor

Affiliation: Medi-Caps University

External Examiner

Name:

Designation

Affiliation

Declaration

I hereby declare that the project entitled **"Virtual Mouse Using Hand Tracking"** submitted in partial fulfillment for the award of the degree of Bachelor of Technology in 'Computer Science and Engineering' completed under the supervision of **Dr. Harsh Pratap Singh, Assistant Professor, Computer Science and Engineering,** Faculty of Engineering, Medi-Caps University Indore is an authentic work.

Further, I declare that the content of this Project work, in full or in parts, has neither been taken from any other source nor have been submitted to any other Institute or University for the award of any degree or diploma.

Nishita Jain

27/04/2023

Certificate

I, **Dr. Harsh Pratap Singh** certify that the project entitled **"Virtual Mouse using Hand Tracking"** submitted in partial fulfillment for the award of the degree of Bachelor of Technology by **Nishita Jain** is the record carried out by them under my guidance and that the work has not formed the basis of award of any other degree elsewhere.

_____ _____

Dr. Harsh Pratap Singh

Computer Science and Engineering

Medi-Caps University, Indore

_____

Dr. Ratnesh Litoriya

Head of the Department

Computer Science & Engineering

Medi-Caps University, Indore

Offer Letter of the Internship

DocuSign Envelope ID: 563FCDC8-8C1A-42F6-B3CC-54D99794DB05

intellicus

Date: 22-Feb-2023

To,
Nishita Jain
1395 Sector D Sudama Nagar
Sudama Nagar - 452009

Dear **Nishita Jain**,

Pursuant to our discussions, **Intellicus Technologies Pvt. Ltd.** is pleased to make you an offer of employment as **Project Trainee** at grade **X** starting on **24-Feb-2023.**
Your Stipend will be INR **15,000**/- per month.

All terms and conditions of your engagement will be in accordance with this letter read along with (i) Annexure A to this letter, (ii) the Appointment Letter (to be issued to you on successful completion of your Training and fulfillment of all documentation requirement), ((iii) the Non- Disclosure Agreement. Formats / copies of the above-mentioned documents are available with us and you are required to read and understand the same prior to your acceptance of our offer for training engagement. In addition, you shall also be required to abide by all the policies and procedures of the Company including those provided on the Company's intranet.

You will be eligible to join as **NA** at Grade **NA** only after successful completion of your Post-graduate degree and on submission of the pass certificate, your CTC will be revised to **NA** .

Please note that this offer is conditional upon satisfactory feedback from your references and necessary background, academic, medical, credit/financial and criminal checks. Our offer is also contingent upon your full and complete disclosure to the Company of any and all agreements (non-competition, non-solicitation, employment, confidentiality or otherwise) with any prior employer, clients, principals, partners or others which in any way limit you either contractually or otherwise from engaging in any business activities required or contemplated by the Company in this offer of training engagement. The Company reserves the right to withdraw this offer of training engagement (for training) without any obligation whatsoever, if it determines or believes that any contractual or other obligation may materially limit your ability to engage in business activities for the Company.

The Company reserves the right to withdraw this offer of employment without any obligation whatsoever, in the event that it presume or suspect or determines or believes that any commercial or

Intellicus Technologies Pvt Ltd
CIN: U31200MP2004PTC016665
SEZ - INDORE Unit I, 4-C, Third Floor, STP-II, Crystal IT Park, Ring Road, Indore, 452001, (M.P.) INDIA
Regd. Office: 1st Floor, Sarda House, 24-B, Palasia, A.B. Road, Indore-452001 (M.P.) India.
Phone: +91.731.4267669, Fax: +91.731.4071256

Completion Certificate

intellicus

26 April 2023

To whomsoever it may concern

This is to notify that Ms. Nishita Jain, pursuing Bachelor of Technology from Medi-Caps University, Indore is undergoing training in Intellicus Technologies India Pvt. Ltd. from February 24, 2023. The training period will end on August 23, 2023.

She is training with Kyvos team under the guidance of Mr. Piyush Jhawar, Lead Quality Engineer.

Sincerely,

For Intellicus Technologies Pvt. Ltd.

Authorized HR Signatory

Intellicus Technologies Pvt. Ltd.
CIN - U31200MP2004PTC016665

SEZ - INDORE Unit I,
4-C, Third Floor, STP-II,
Crystal IT Park SEZ,
Ring Road, Indore-452001
(M.P.) INDIA
Fax : +91.731.4743111

Regd. Office :
1st Floor, Sarda House, 24-B,
Palasia, A.B. Road, Indore-452001
(M.P.) INDIA
Tel : +91.731.4067669
Fax : +91.731.4071256

www.intellicus.com

Acknowledgements

I would like to express my deepest gratitude to Honorable Chancellor, **Shri R C Mittal,** who has provided me with every facility to successfully carry out this project, and my profound indebtedness to **Prof. (Dr.) Dilip K. Patnaik,** Vice Chancellor, Medi-Caps University, whose unfailing support and enthusiasm has always boosted up my morale. I also thank **Prof. (Dr.) Pramod S. Nair,** Dean, Faculty of Engineering, Medi-Caps University, for giving me a chance to work on this project. I would also like to thank my Head of the Department **Dr. Ratnesh Litoriya** for his continuous encouragement for betterment of the project.

I express my heartfelt gratitude to my **External Guide, Mr. Piyush Jhawar**, Project Lead, Intellicus Technologies Pvt. Ltd as well as to my Internal Guide, **Dr. Harsh Pratap Singh,** Assistant Professor, Department of Computer Science Engineering, MU, without whose continuous help and support, this project would ever have reached to the completion.

I would also like to thank to my team at Intellicus Mr.Chirag Gupta, Mr. Sanchit Agnihotri, Mr. Jayesh Gupta, Ms. Naina Pandey, Mr. Mayank Godiyal, Ms. Paridhi Jain who extended their kind support and help towards the completion of this project.

It is their help and support, due to which we became able to complete the design and technical report. Without their support this report would not have been possible.

Nishita Jain (EN19CS301227)
B.Tech. IV Year
Department of Computer Science & Engineering
Faculty of Engineering
Medi-Caps University, Indore

Executive Summary

Virtual Mouse using Hand Tracking is an Artificial Intelligence-based Desktop-based application, developed in Python 3.8 that aims to eliminate the need for physical pointing devices such as an optical mouse or wireless mouse by the means of hand gestures. It is very easy to operate, as it needs only a Webcam with a suitable resolution (0.9 Megapixel at least) and a 60 Hz display. Our software takes the input captured by this webcam using the OpenCV library, processes the input to identify the target hand (for our project we are considering only a single hand) using the MediaPipe library. After identifying the target hand, it detects the gestures made by the user and tries to recognize them from the predefined gesture. As soon as a predefined gesture is identified, it switches the mode from the existing one to which the user is currently pointing. After switching the mode, it performs the predesignated operation using PyAutoGui and AutoPy library. The application exits as soon as the user presses ‘q’ on their keyboard (quit).

Keywords:

Artificial Intelligence, Hand Gestures, OpenCV, MediaPipe, PyAutoGui, AutoPy

Table of Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION

1.1 Introduction

It has been generations since we have been using hand gestures for communicating in human the society. So then why not apply it to the machines that we are using.

We are living in the era of Artificial Intelligence, most of the things around us are either automated or replaced but still, mouse remained the same, the computer mouse that we are using today isn't that much different which was first created in 1964 except being wireless and optical.

Most of the laptops and other devices require either touch or mouse interface to work, imagine you are reading a book or watching a movie sitting far from your laptop/desktop, and for operating you have to go close to the device or buy a wireless mouse (which is going to charge you good amount), won't it be easy to action our hand and the appropriate task will be done.

The evolution of the User Interface (UI) witnessed the development from text-based UI based on the keyboard to graphical UI based on mice. In current virtual environments applications, keyboards, mice, and joysticks are still the most popular and dominant devices.

With the development of augmented-reality technology, researchers are working to reduce people's workload while increasing their productivity by studying human-computer interactions (HCI). The Natural User Interface (NUI) of hand-gesture recognition is an important topic in HCI. Hand-gesture-based interfaces allow humans to interact with a computer in the most natural way, typically by using fingertip movements. Fingertip detection is broadly applied in practical applications, e.g., virtual mice, remote controls, sign-language recognition, or immersive gaming technology. Therefore, virtual mouse control by fingertip detection from images has been one of the main goals of vision-based technology in the last decades, especially with traditional red-green-blue (RGB) cameras. Fingertip detection with multiple people simultaneously poses a great difficulty that current systems have not yet overcome.

1.2 Literature Review

So, there is been a lot of work on the topic of the virtual mouse in recent times. Many have designed their applications and some of them are:

S.No	Authors	Abstract	Year
1.	D.L. Quanm	An experiment was conducted to investigate gesture recognition with a human hand manipulating the Data Glove, an electronically instrumented glove which provides information about finger and hand position. A total of 22 gestures in three classes were investigated. The first class contained gestures which only involved finger flexure. The second class contained gestures which required both finger flexure and hand orientation. The third class of gestures required finger motion in addition to flexure and orientation. Only four sensors were necessary to positively identify specific gestures from groups of up to 15 gestures. The results show the specific number of sensors required to positively identify a gesture from a group. This depends on the number of gestures in a group, as well as the class of gestures.	2002
2.	C.-C. Hsieh, D.-H. Liou, and D. Lee	Hand gesture recognition based man-machine interface is being developed vigorously in recent years. Due to the effect of lighting and complex background, most visual hand gesture recognition systems work only under a restricted environment. An adaptive skin color	2010

		model based on face detection is utilized to detect skin color regions like hands. To classify the dynamic hand gestures, we developed a simple and fast motion history image based method. Four groups of haar-like directional patterns were trained for the up, down, left, and right hand gestures classifiers. Together with fist hand and waving hand gestures, there were totally six hand gestures defined. In general, it is suitable to control most home appliances. Five persons doing 250 hand gestures at near, medium, and far distances in front of the web camera were tested. Experimental results show that the accuracy is 94.1% on average and the processing time is 3.81 ms per frame. These demonstrated the feasibility of the proposed system.	
3.	N. P. S. Angel	They created a useful framework for real-time gesture recognition that can be used in a range of human-computer interaction applications. But, it was unable to work at a complex background and was computable only under good light.	2013
4.	A.Haria, A. Subramanian, N. Asokkumar, S. Poddar, and J. S. Nayak	The technique of building a process of interaction between human and computer is evolving since the invention of technology. The mouse is a superb invention in HCI (Human-Computer Interaction) technology. Though wireless mouse technology is still being	2017

		invented still, that technology isn't completely device free. A Bluetooth mouse needs battery power and a connecting dongle. The proposed mouse system is beyond this limitation. This paper proposes a virtual mouse system supported by HCI using computer vision and hand gestures. Gestures captured with a built-in camera or webcam and processed by a Convolutional Neural Network Model for classification among the desired mouse operations. The users are going to be allowed to regulate a number of the pc cursor functions with their hand gestures. Primarily, a user can perform left clicks, right clicks, and double clicks, scrolling up or down using their hand in several gestures. This technique captures frames employing a webcam or built-in cam and processes the frames to make them track-able and then recognizes different gestures made by users and perform the mouse functions. Therefore the proposed mouse system eliminates device dependency so as to use a mouse.	
5.	N. Waybhase, H. Joshi, R. Litoriya, and D. Mangal	We tend to notice a lot of development in the computing world in today's modern world. Computer science is a technique that combines today's technologies. This paper is also backed by a small portion of AI. This paper shows how to use our computer's window exploitation camera to create a hand	2022

		gesture-based virtual mouse that allows you to control the entire system by merely moving your fingertips. Finger detecting methods for quick camera access and simple computer software make it even more accessible. A motion tracking mouse is implemented using the system. This technology eliminates the need for a physical mouse, saving time and effort.	

Table 1.2.1: Literature Review

1.3 Objective

To overcome the problem stated above the need for a virtual mouse was raised. In this project, we are designing an application where the working of the mouse can be mimicked by using a hand-gesture recognition system. We propose an open-source virtual mouse, which will make us feel connected while using our computer devices, an all-new experience of getting things done by just use of our hands. We are aiming in creating a cost-free AI-based approach for hand recognition software for laptops and PCs with webcam support. The project covers a hand recognition tool, which is used to move the mouse pointer, perform simple operations like clicking and dragging.

1.4 Significance

The proposed application will help eliminate the cons of existing pointing devices, such as the need for a physical medium between the display and the pointing device also our application proposes a contactless pointing device therefore the chances of hardware damages are less.

There is no liability of using color buttons or tapes around the tip of fingers for the identification of hand gestures for performing different actions, unlike other related works that use color strips to map the functionality of the mouse button to the tip of the fingers.

There is no need for multiple cameras to detect or record the hand gestures for implementing the functionalities to the cursor a single camera with adequate quality and good lighting is enough for the application.

This tool can help detect complex hand motions to perform various other functionalities such as building a virtual keyboard or a finger counter or other hand tracking applications.

1.5 Physical Mouse Difficulties

1. Inappropriate drivers- The system drivers are one of the first ports of call for a Windows mouse issue. Windows takes care of the driver updates for most of the hardware but sometimes it is not possible to find the drivers which causes difficulties in using the mouse.

2. Mouse Freezing and Disappearing Cursor- Common and frequently occurring mouse problems include frozen mouse pointers, double-clicking issues, not working mouse, uneven mouse speed, and uneven mouse pointer behavior. Their fixes include cleaning the mouse, checking the connections, reinstalling the mouse driver software, and many more.

3. Mouse Lagging and Beeping Sound- It usually happens when scrolling a mouse cursor with a wireless mouse. This situation is very common when switching to different operating systems. Mouse stuttering or lag is mainly caused by mouse driver, outdated graphics card, internet problems, etc.

4. Frozen Touchpad- Mostly while using touchpads the user faces many problems such as driver issues and software updates. In most of the cases due to rough and bad handling the touchpad stops working and the user needs to use a physical mouse for performing further actions.

5. The wired mouse cannot be used in long range as the ranges are limited for better connectivity.

6. The wired mouse is delicate in nature and must be used carefully as it can be damaged easily.

7. If the batteries are low they tend to lose their accuracy and speed.Mouse heavily relies on batteries. Also they work on flat surfaces.

8. Wireless mice can sometimes face connectivity issues.

Chapter 2

REQUIREMENT ANALYSIS

2.1 Product Description

Virtual Mouse using Hand Tracking is a desktop-based application where a user can mimic the basic functionalities of a mouse by simply the gestures using his hands. The application uses Artificial Intelligence and Machine Learning algorithms to extract features from hand gestures that are captured using a simple webcam. It eliminates the need for external pointing devices such as a mouse or joystick and provides precision and ease of access. It is based on Python3 and libraries include OpenCv, MediaPipe, PyAutoGUI, and AutoPy.

2.2 Functional Requirements

- **Receive Captured Video:**
    - The program must be able to receive the captured video stream from the web camera of the operating machine
- **Target Hand Detection:**
    - The program must be able to detect the hand of the target user and draw a bounding box around it so as the motion of the hand could be separately observed from what is in the background
- **Hand Landmark and Action Bounding Box Identification:**
    - After identification of the hand of the target user, the program must be able to identify the hand landmarks i.e. fingers palm so as distinguish between its movement.
    - The abounding box that acts like a mouse pad must be visible to the user so that the hand doesn't move out of the captured frame.
- **Hand Tracking:**
    - The program can mark and identify the motion of the hand along with the hand landmarks as they move through the captured frame.

- **Gesture Recognition:**
    - The program must be able to distinguish between the different hand gestures and switch from moving mode to clicking/selection mode when the hand gesture is changed in the course.
- **Display processed video:**
    - The program must be able to display the real-time processed video of the input video stream for the sake of user convenience.

2.2.1 Hardware Requirements

The program can run on minimum hardware requirements, however:

- Intel Dual Core processor or later.
- Minimum 2 GB RAM
- 60 Hz display
- Web camera of quality greater than VGA (0.9 MP)
- Hard disk 200 GB.

is recommended.

2.2.2 Software Requirements

Basic Requirements include:

- Windows 7 operating system or above is required.
- Webcam driver
- Python V 3.6 or above.

2.3 Non-Functional Requirements

- Allowed Access: The user's video will only be captured only when the application is launched with the user's permission no privacy breach will be caused.
- Maintenance Requirements: There could be some new modules or update within the existing modules to improve the performance of the application.
- Portability: Desktop Based Applications can be used on machines fulfilling the hardware and software requirements.
- Performance: The change in input won't affect the processing time.

2.4 Feasibility Study

• Technical Feasibility:

The hardware requirements consider only a webcam that is economical and could be managed for the project purpose. Also, the software required is open-source and the libraries used have proper documentation.

• Financial Feasibility:

The user has to purchase an external webcam (if not available) which doesn't cost much and can be easily managed.

• Risk Feasibility:

Proper exposure to the project might not be possible since it is implementing only basic functionalities of a mouse, there is a possibility that we might not get a proper response which can affect the improvement and performance of the software.

2.5 Technology Description

- **Python 3–**

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. Python 3.0 was released in 2008.

- **OpenCV –**

It is a huge open-source library for 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.

- **MediaPipe –**

MediaPipe offers open-source cross-platform, customizable ML solutions for live and streaming media. It provides a solution in Android, iOS, Python, and JS such as Hand Tracking, Face Mesh, etc.

- **PyAutoGui–**

Pyautogui is a library that allows you to control the mouse and keyboard to do various things. It is a cross-platform GUI automation Python module for human beings. As it is a third-party library, we need to install it.

- **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. Currently supported on macOS, Windows, and X11 with the XTest extension.

Chapter 3

DESIGN ANALYSIS

3.1 UML Diagrams

The Unified Modeling Language is a general-purpose, developmental, modeling language in the field of software engineering that is intended to provide a standard way to visualize the design of a system.

3.1.1 Use Case Diagram

Use-case diagrams model the behavior of a system and help to capture the requirements of the system. Use-case diagrams model the behavior of a system and help to capture the requirements of the system.

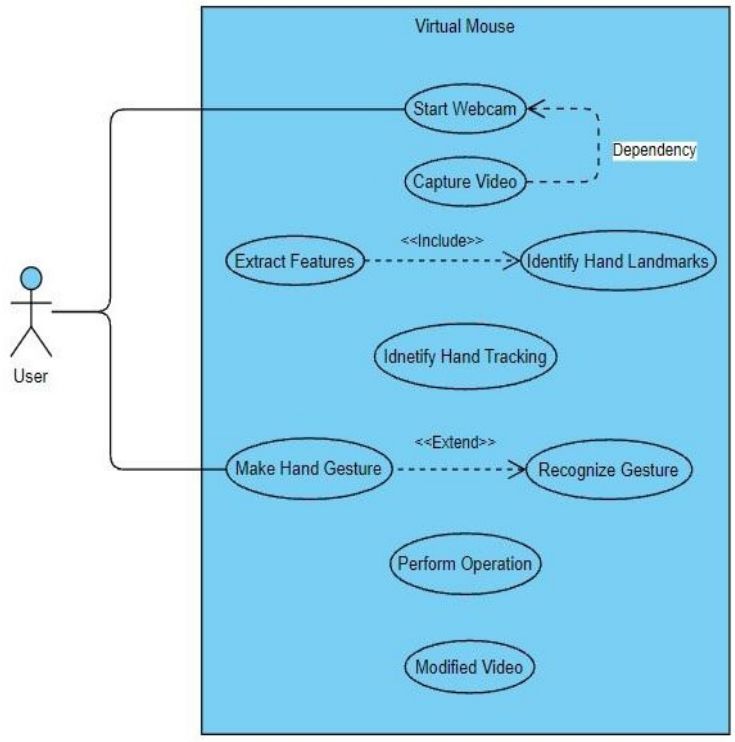

Figure 3.1.1 Use Case Diagram

Actors: User

System: Virtual Mouse

Use Cases: Start Webcam, Capture Video, Extract Features, Identify Hand Landmarks, Identify Hand Tracking, Make Hand Gesture, Recognize Gesture, and Perform Operation, Modified Video.

3.1.2 Activity Diagram

An activity diagram is a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.

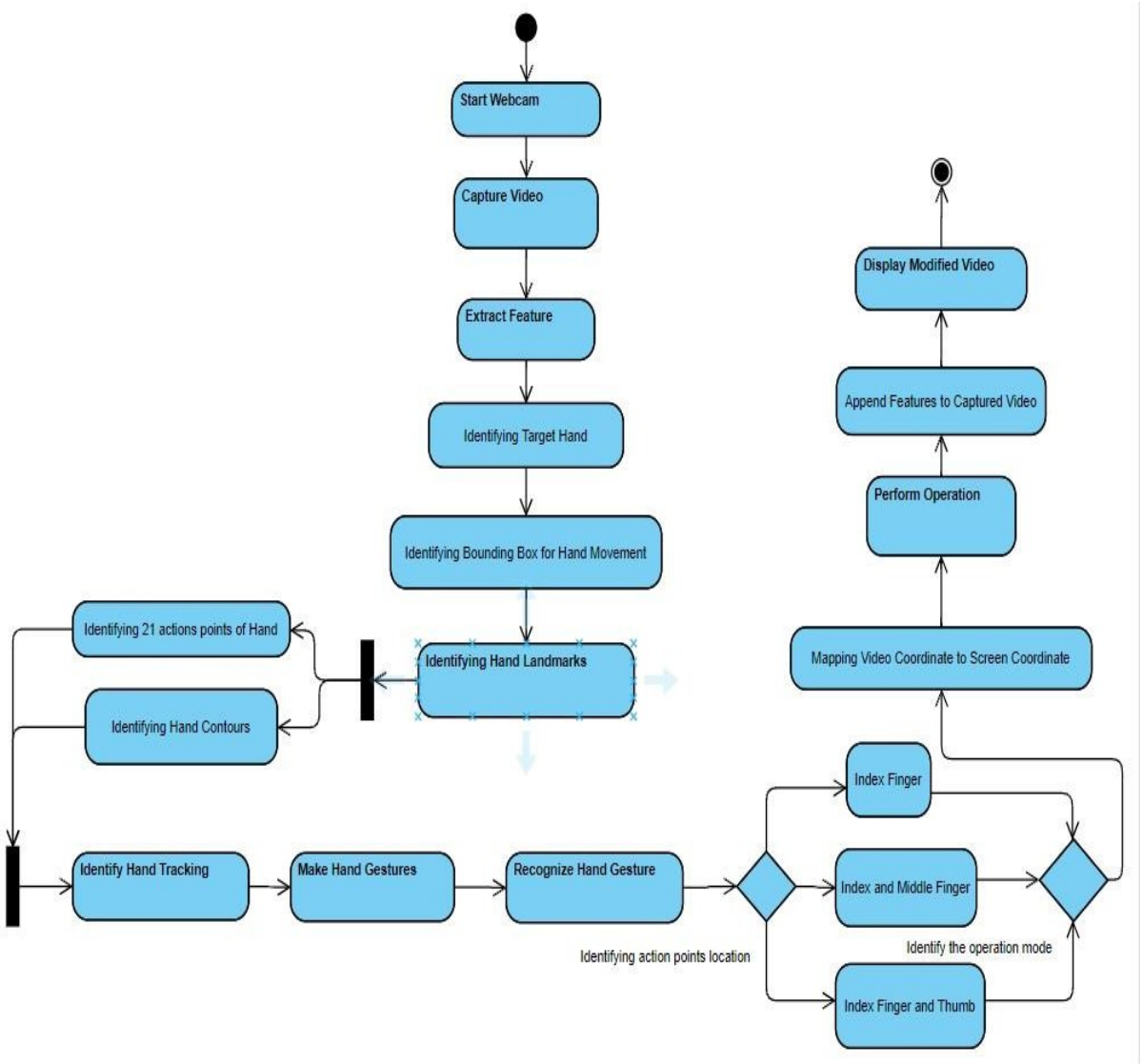

Figure 3.1.2 Activity Diagram

Activities:

Start Webcam, Capture Video, Extract Features, Identify Hand Landmarks, Identify Hand Tracking, Make Hand Gestures, Recognize Hand Gestures, Perform Operation, Display Modified Video.

3.1.3 Class Diagram

The class diagram depicts a static view of an application. It represents the types of objects residing in the system and the relationships between them.

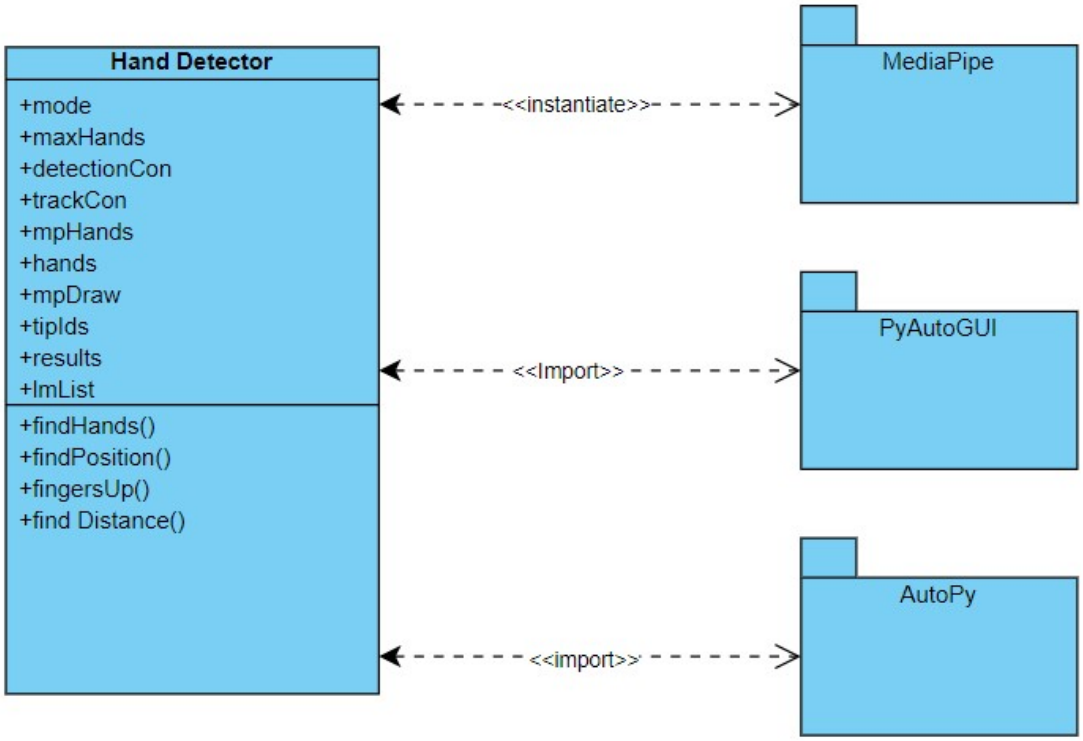

Figure 3.1.3 Class Diagram

Class: Hand Detector (Control Class)

Packages: MediaPipe, PyAutoGUI, AutoPy

Relationships:

1. Instantiate relationship between Hand Detector class and MediaPipe package.
2. Import relationship between Hand Detector class and PyAutoGUI package.
3. Import relationship between Hand Detector class and AutoPy package.

3.1.4 Sequence Diagram

A sequence diagram or system sequence diagram (SSD) shows object interactions arranged in time sequence in the field of software engineering. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

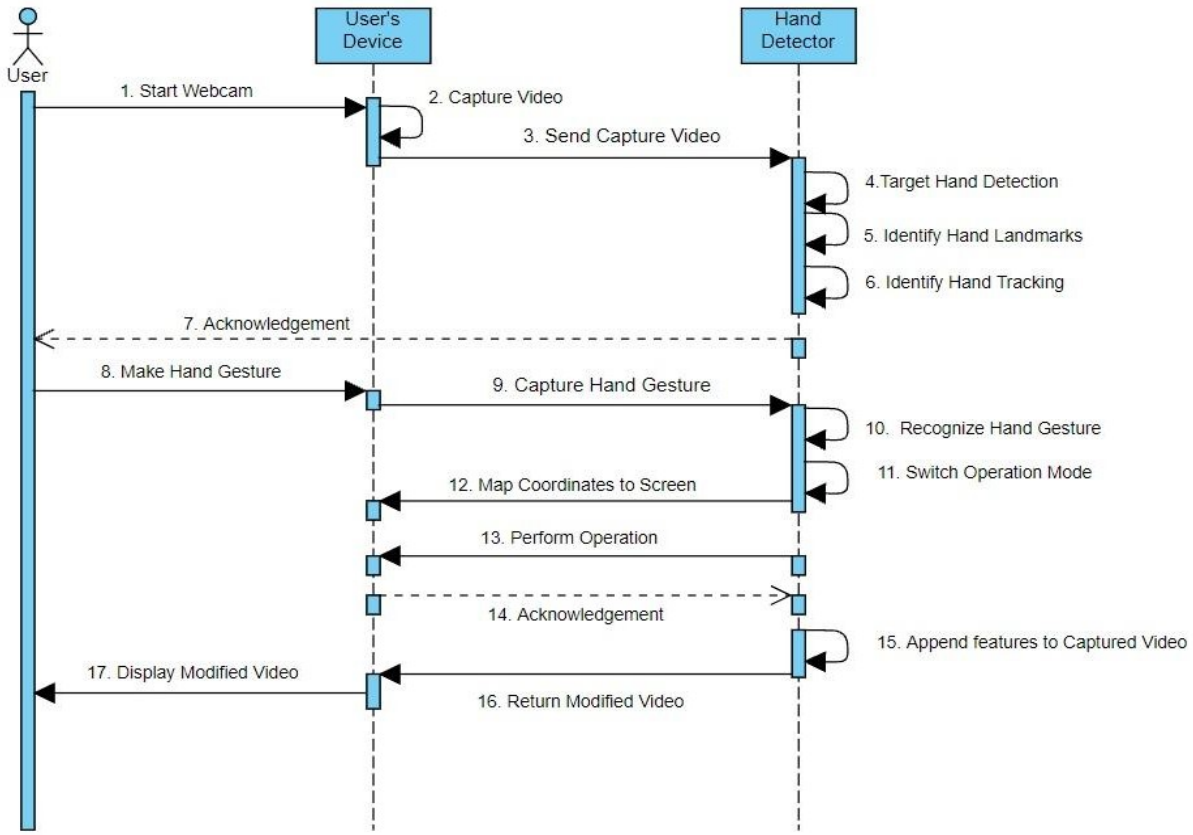

Figure 3.1.4 Sequence Diagram

3.2 Hand Gestures

Hand gestures are an integral part of communication. Virtual Mouse is based on human-computer interaction that is based on detection of a user's hand gestures to perform the basic operation of a mouse.

- Controlling mouse movements using pyautogui module: Python tracks and controls mouse using the coordinate system of the screen. Suppose the resolution of your screen is 1920X1080, then your screen's coordinate system looks like this:
- size(): This function is used to get Screen resolution.
- moveTo(): use this function to move the mouse in the pyautogui module. This function takes x and y coordinates, and an optional duration argument. This function moves your mouse pointer from its current location to x, y coordinate, and takes time as specified by duration argument to do so.
- Binary\Gesture Encoding:

  FIST = 0

  PINKY = 1

  RING = 2

  MID = 4

  LAST3 = 7

  INDEX = 8

  FIRST2 = 12

  LAST4 = 15

  THUMB = 16

  PALM = 31

Using the MediaPipe library we can identify hand landmarks as shown in figure 3.2.1 below:

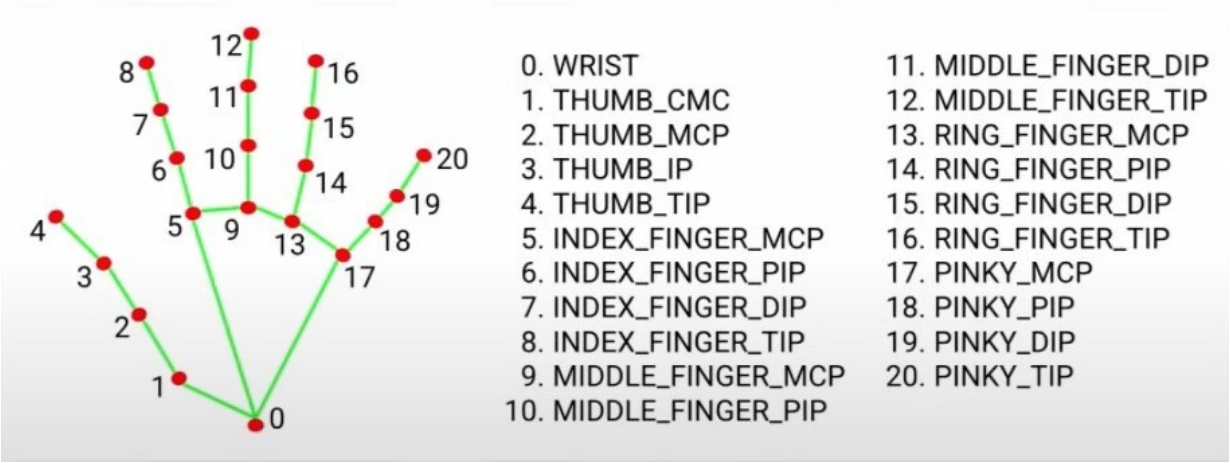

Figure 3.2.1 Identified Hand Landmarks

Using these Hand Landmarks we have identified certain hand gestures for the Virtual Mouse to perform the basic operations of a mouse. These gestures are depicted below:

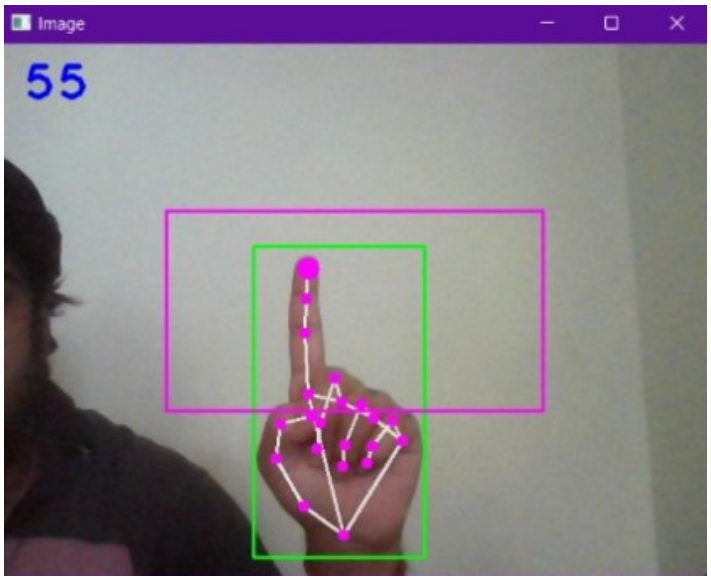

Figure 3.2.2 Cursor Drag Gesture

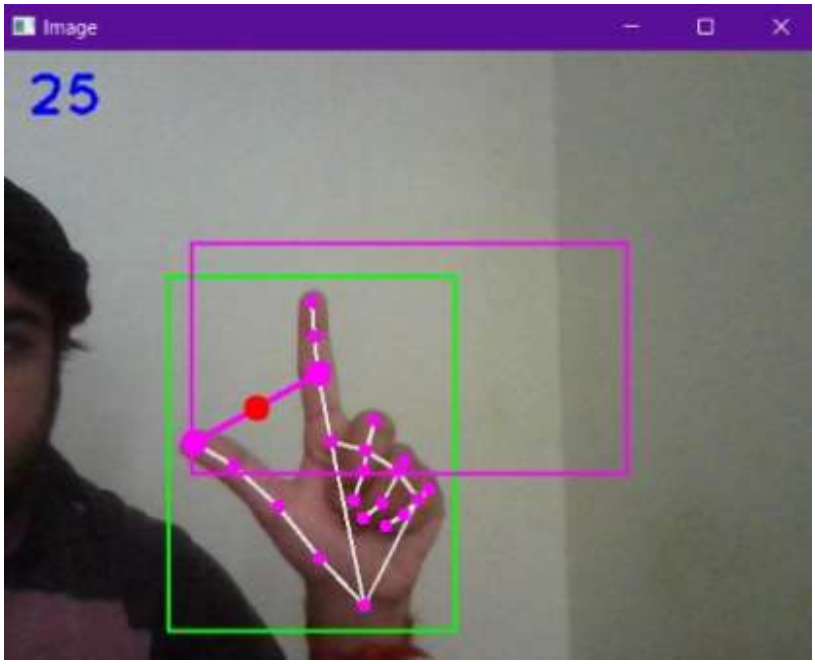

Figure 3.2.3 Left Click Gesture

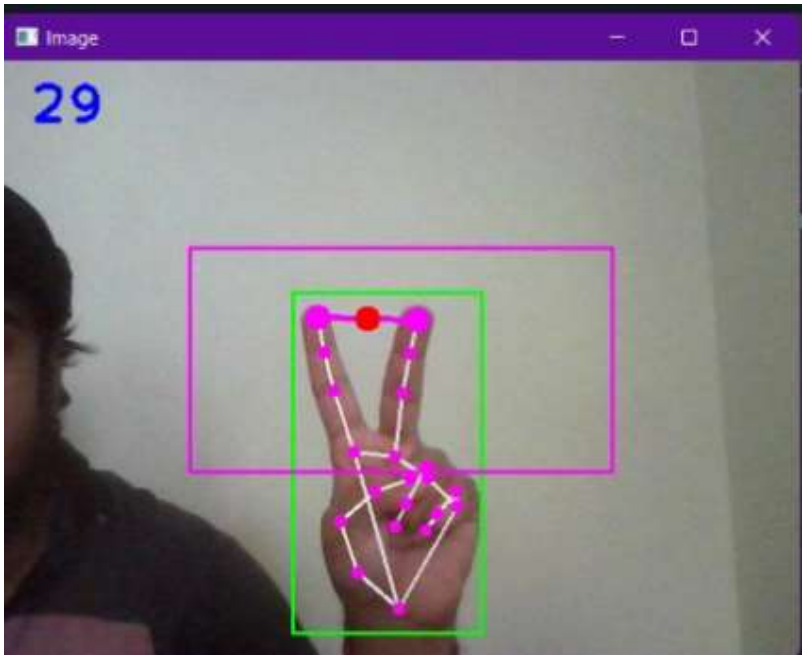

Figure 3.2.4 Right Click Gesture

Chapter 4

ALGORITHM

PYCharm Libraries for Hand Gesture Detection

- cv2: OpenCV is an open-source computer vision library used for various computer vision tasks such as image and video processing, object detection, and recognition.
- mediapipe: A Google library for building machine learning pipelines to process multimedia content such as images, videos, and audio.
- pyautogui: A cross-platform GUI automation library that enables the script to control the mouse, keyboard, and other system functionalities.
- math: A built-in library for mathematical operations.
- enum: A built-in library for creating enumerated constants.
- ctypes: A Python library to create, manipulate and call C functions in Python.
- comtypes: A Python package for using and creating COM objects.
- pycaw: A Python package that provides an interface to control the Windows Audio Session API (WASAPI).
- google.protobuf.json_format: A Python library for converting Protocol Buffers messages to and from JSON format.
- screen_brightness_control: A Python package to control the screen brightness on Windows, Mac, and Linux.

More than these above-specified libraries, The AI virtual mouse system makes use of the transformational algorithm, and it converts the coordinates of fingertips from the webcam screen to the computer window full screen for controlling the mouse.

Functions And Classes used for Mouse Controls

__init__() - Constructor method that initializes the HandRecog class with necessary variables.

update_hand_result(hand_result) - Method that updates the HandRecog class with the current hand results.

get_signed_dist(point) - Method that calculates the signed distance between two points on the hand landmarks.

get_dist(point) - Method that calculates the distance between two points on the hand landmarks.

get_dz(point) - Method that returns the absolute difference in z-coordinates between two points on the hand landmarks.

set_finger_state() - Method that updates the binary-encoded gesture based on the current finger state.

get_gesture() - Method that returns the recognized gesture based on the current binary-encoded gesture.

__init__(hand_label) - Constructor method that initializes the Controller class with necessary variables.

mouse_move(x, y) - Method that moves the mouse pointer to the specified (x, y) coordinates.

mouse_drag(x, y) - Method that simulates dragging the mouse pointer to the specified (x, y) coordinates.

mouse_click(button) - Method that simulates clicking the mouse button specified by button.

mouse_scroll(amount) - Method that simulates scrolling the mouse wheel by the specified amount.

keyboard_press(key) - Method that simulates pressing the specified key on the keyboard.

volume_up() - Method that increases the system volume.

volume_down() - Method that decreases the system volume.

brightness_up() - Method that increases the screen brightness.

brightness_down() - Method that decreases the screen brightness.

execute_gesture(gesture) - Method that executes the appropriate action based on the recognized gesture.

run() - Method that continuously processes the hand landmarks and executes actions based on the recognized gesture.

Step	Description	Function
1	Import Libraries	‘cv2’, ‘numpy’, ‘math’, ‘pyautogui’
2	Define variables and parameters	‘bg’ (background image), ‘bg_avg’ (average of background image), ‘threshold’ (threshold for detecting movement), ‘area_threshold’ (minimum area of contour to be considered), ‘learning_rate’ (rate at which the background image is updated), ‘prev_x’ and ‘prev_y’ (previous x and y positions of the mouse), ‘smooth_factor’ (smoothing factor for mouse movement)
3	Capture initial background image	cv2.imread(), cv2.cvtColor(), cv2.GaussianBlur(), cv2.accumulateWeighted(), cv2.convertScaleAbs()
4	Process each frame from the camera	v2.cvtColor(), cv2.absdiff(), cv2.threshold(), cv2.dilate(), cv2.findContours(), cv2.drawContours()
5	Find largest contour (hand) and calculate its centroid	‘cv2.contourArea()’, ‘cv2.moments()’
6	Smooth out mouse movement	‘smooth_factor’ and linear interpolation
7	Move mouse to new position	‘pyautogui.moveTo()’

Table 4.1 Main Steps and Functions Used in Gesture Recognition Virtual Mouse

Virtual Mouse using Hand Tracking is a desktop-based application currently functioning for Windows Operating System. This application has a defined set of operations and functionalities that happen in a defined flow. This flow of operations can be easily understood from an algorithm that is:

Step 1. Perform Step 2 to Step 10 until the user press ‘Q’ on the keyboard.

Step 2. Capture the video from the user’s web camera.

Step 3. Initialize an object of MediaPipe Hands Class and process the input video frame.

Step 4. If a hand is captured in the defined frame of the video then identify the Hand Landmarks respective to the Hand and draw them.

Step 5. Draw a bounding box for the user's hand and store the position coordinates of the Hand Landmarks.

Step 6. Identify the Handedness (Left Hand or Right Hand) and check the position of the fingers and thumb, and then identify a gesture.

Step 7. In case of only the Index finger is up and Thumb and Middle finger are down then it is a cursor drag. Map the coordinates of the reduced frame to the screen coordinate, smoothen the values of calculated coordinates, and then use the AutoPy library to perform cursor drag operation.

Step 8. In case of both Thumb and Index Finger is up, calculate the distance between the tips and check it against a threshold if the value is less than the threshold performs a Left-Click operation.

Step 9. In case of both Index and Middle finger up, calculate the distance between the tips and check it against a threshold if the value is less than the threshold performs a Right-Click operation.

Step 10. Calculate the FPS and display the captured video after appending the features until the terminating condition is reached.

Chapter 5

RESULT AND DISCUSSIONS

5.1 Results

The project was able to fulfill all the listed functional and non-functional requirements. After the completion of our project, we subjected it to testing and found out that the listed operations were performed efficiently the results of which could be demonstrated with the help of the following figures:

Figure 5.1.1 shows that the application can identify the target hand and draw hand landmarks (with pink circles), hand connections (white lines), and bounding box (green rectangle) around it, also displays the calculated FPS.

Figure 5.1.1 Identifying the Target Hand

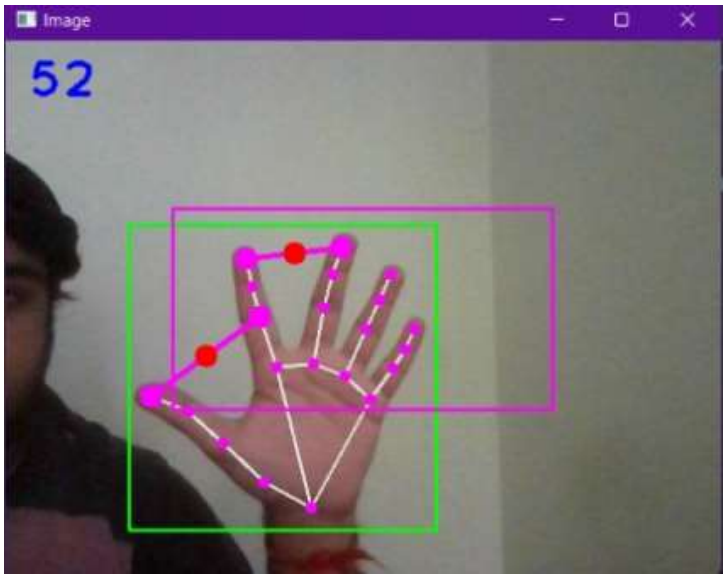

Figure 5.1.2 Displaying Reduced Windowpane for the operation of Hand Gesture

Figure 5.1.2 displays the application offers the reduced frame (purple rectangle) for the movement of the Hand so that there is no discontinuity in hand detection. It also displays the Hand points that are to be used for gesture-based operation.

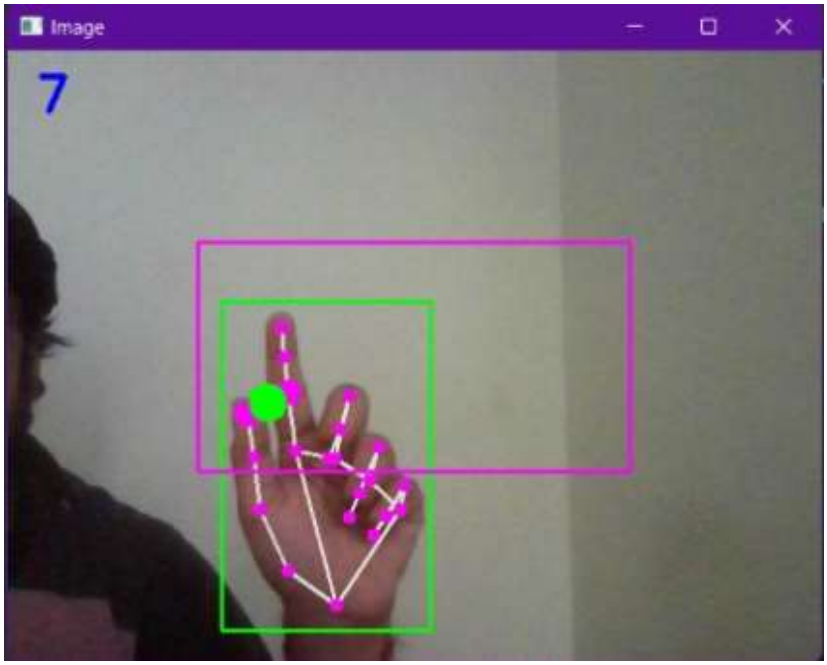

Figure 5.1.3 Left –Click Executed

Figure 5.1.3 shows the execution of the Left-Click using the Hand gesture, as stated in the algorithm as the distance between the tips of the Thumb and Index finger is less than a certain threshold
length (45 pixels) the operation gets executed.

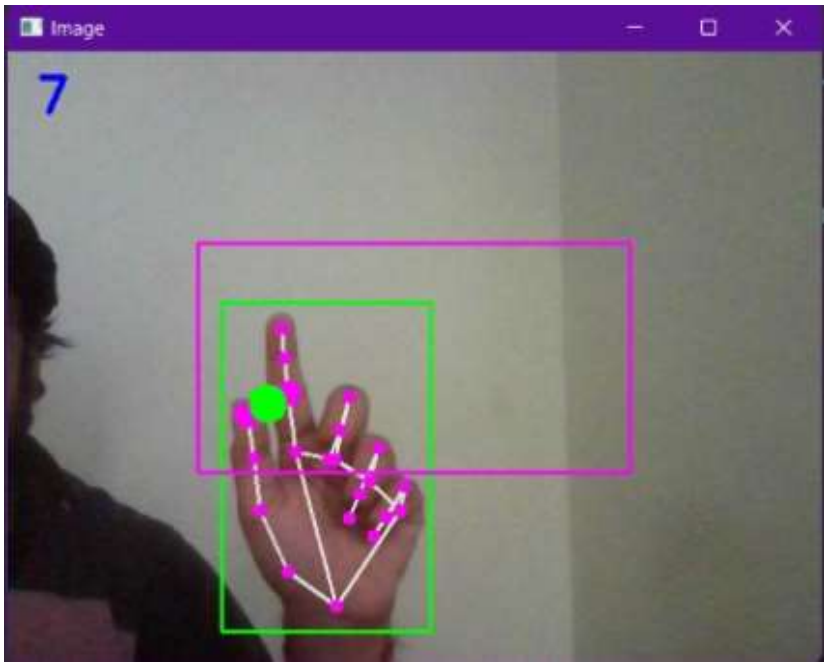

Figure 5.1.4 Right –Click Executed

Figure 5.1.4 shows the execution of the Right-Click using the Hand gesture, as stated in the algorithm as the distance between the tips of the Index finger and Middle finger is less than a certain threshold length (35 pixels) the operation gets executed.

5.2 Discussions

The object of MediaPipe library Hands class takes certain parameters as input before creating the object for Hand Detection these parameters are: number of hands, static image mode, detection confidence, and tracking confidence. Here, the number of hands identifies the maximum number of hands to be detected in the captured, which for our project is set to '1' as we are currently focused on single hand gestures.

Static image mode that corresponds to how the model will process the images, for our project it is set to 'False' that means the model will try to detect the hands in the first input image,

and once the maximum number of hands is detected it will identify the corresponding hand landmarks and localize them without invoking further hand detections until the detected hands in the frame are less than the maximum number of hands.

Detection Confidence is the value from the hand detection model that symbolizes the probability that the current frame contains a hand, for our project purpose we have set it to 0.75. Tracking Confidence is the value from the landmark-tracking model that is used to identify the hand landmarks for the detected hand, for our project we have set it to 0.5.

The model performance and robustness are quite dependent on what are the values of detection confidence and tracking confidence. Too high value for detection confidence will try to identify a perfect hand geometry in the captured frame that will ultimately be a trade-off in case of cheap hardware.Too high a value of tracking confidence will require more processing time or higher latency whereas too low a value might make unnecessary landmark detections.

Not only these parameters but the background and lighting in the captured frame also affect the results. Too much complex background might induce unnecessary detections that would decrease model efficiency. Improper lighting such as too much light or dark room might not be the places for the ideal operation of our application.

The hardware and software requirements do not cover up the altering effects of the lighting, background complexity, model parameters, etc. For this project, the user is not allowed to interfere with the model parameters as of now we determine the best-suited values using testing. The stated specifications for the hardware and software can deal with mild changes in the background and lighting however unfavorable situations are still a challenge for the project.

Chapter 6

SUMMARY AND CONCLUSION

6.1 Summary and Conclusion

Virtual Mouse using Hand Tracking is a project of basic need. As in the evolving times, people are demanding smart alternatives for existing problems and this project is just covering up the need for replacing physical pointing devices. Artificial Intelligence techniques help users to operate their devices with more ease and comfort. This software will eradicate issues with a physical mouse such as the requirement of the flat surface, additional driver installation or compatibility issues, physical damages, etc., and offer the user a fresh experience. However, this software provides just the bare minimum operations of a mouse and its feature can be enhanced.

This Project Report discusses what are the current scenarios and the previous work done in the field and highlights that there is no open to all applicants for the users. Software description and the Feasibility along with Functional and Non-Functional requirements for the proper functioning of the software along with the technologies that are required to implement the software are described. The theoretical implementation necessary for the design of class, activity flow, and algorithm is listed along with their diagrams and a brief description. The Algorithm along with the listed hand gesture for the functioning of the software are also included within the report. Results obtained after the completion of the project and its conversion to a desktop application are discussed with figures and discussion about the affecting parameters are also included for the understanding. The future scope and ideas related to the extension of the project are also discussed for future references.

Chapter 7

FUTURE SCOPE

The current project is sufficient in fulfillment of designing a virtual mouse that performs the basic mouse operations, but it is inefficacious to perform complex operations such as:

- Click and drag:

  When a user wants to select multiple items/files, this can be achieved by holding either right or left click and dragging the cursor over the desired files it can be selected.

- Drag and drop:

  If the user wants to move a particular or a group of files/items from one directory to another or to rearrange the same directory, the items can be selected and moved by holding the left click button to operate.

- Scroll:

  The modern mouse is equipped with a scroll wheel which is used to operate either scroll the window up or down.

- Accuracy and Precision for detection of target hand can be improved for too dim or bright backgrounds.
- Hand Gestures can be modified as per convenience for better usability to increase user-friendliness.
- The operability of Hand Gestures could be fine-tuned for varying hardware to improve the precision and accuracy of the executed operation.
- The current software works fine with Windows operating system but it could be extended to other operating systems like LINUX, MacOS, etc.
- The current system is based on a pre-trained model of the MediaPipe library for the functionality of Hand Detection which can be improved by training it on a more accurate or improved CNN.