A MINI PROJECT -2 REPORT

ON

"VIRTUAL MOUSE AND KEYBOARD USING CNN"

Submitted to Kakatiya University Warangal

for the partial

fulfilment of the Requirement for the award of degree of

BACHELOR OF TECHNOLOGY IN

COMPUTER SCIENCE & ENGINEERING

Submitted by

MOHAMMAD KARISHMA (21626T0904)

Under the guidance of

Mrs. SHIREEN FATHIMA

(Assistant Professor)

DEPARTMENT OF COMPUTER SCIENCE & TECHNOLOGY

AIZZA COLLEGE OF ENGINEERING AND TECHNOLOGY

(Affiliated to Kakatiya University, Warangal)

Village: Mulkalla, Mandal: Hazipur, District: Mancherial-504 209

AIZZA COLLEGE OF ENGINEERING AND TECHNOLOGY

(Affiliated to Kakatiya University, Warangal)

Village: Mulkalla, Mandal: Hazipur, District: Mancherial-504 209

CERTIFICATE

This is to certify that dissertation entitled

"VIRTUL MOUSE AND KEYBOARD USING CNN"

Submitted by **MOHAMMAD KARISHMA(21626T0904)** in the partial Fulfilment for the requirement of the degree of Bachelor of Technology, Kakatiya University,Warangal

Internal Guide

(C.S.E)

Head of the department

(C.S.E)

External Examiner

Principal

(AZCET)

ACKNOWLEDGEMENT

I am thankfull to our principal **K.SRIRAM,** Aizza college of Engineering & Technology providing the necessary facilities for pursuing the project work.

I extend my sincere and heart full thanks to my **H.O.D. Mrs. SUMERA MOHAMMADI** Aizza College of Engineering and Technology, who had been a constant source of inspiration to me for her motivating in making this project successful.

I am thankful to **Mrs. CH. MALLESHWARI** Assistance Professor of Computer Science Department , Aizza college of Engineering and Technology, who guided me in successfully Completely my project entitled **"VIRTUAL MOUSE AND KEYBOARD USING CNN"**
I am thankful to my Parents and Friends for the moral support troughout the project work That helped to strengthen my will power

MOHAMMAD KARISHMA

(21626T0904)

ORGANIZATIONAL PROFILE

My Institution name **Aizza college of Engineering & Technology** established in the Year 1999, affiliated to Kakatiya University, Warangal in 2019. It is located in outskirts of Mancherial in a peaceful environment. Our college consists of well experienced staff and good infrastructure. The college has the department of CSE, EEE, MINING and MBA. In our department we have a Research of Software development cell equipped with high configuration systems. It was inaugurated in 2010, from which many students got opportunity to do live projects and publish research papers in intrnational journals.

- I have done project on **“VIRTUAL MOUSE AND KEYBOARD USING CNN”**

to control the computer cursor functions with using hands.

ABSTRACT

The keyboard and mouse are one of the most amazing Human-Computer Interaction (HCI) creations. Currently, a wireless mouse and keyboard still use devices and are not free of devices completely since it uses a battery for power and a dongle to connect them to the PC. This issue can be overcome in the proposed virtual keyboard system and mouse by using a webcam or a built-in camera to capture hand motions and recognize hand tips using computer vision.

The proposed model uses a Convolutional neural network (CNN) machine learning algorithm. The computer may be controlled remotely using hand movements and may perform left-click, right-click, scrolling operations, and computer cursor functions and control keyboard keys without the requirement of a hardware mouse and keyboard.

The user will be allowed to control the computer cursor functions with their hands. In the proposed model, we train a model to classify gestures into various alphabets and numbers and we use the trained model for automating the keyboard inputs.

Brightness control with hand detection uses Open CV and Media pipe to see hand gesture and accordingly set the brightness of the system from a range of 0-100. For detecting palms, the method is based on deep learning algorithms like CNN (Convolutional neural network).

CONTENTS

LIST OF FIGURES

CHAPTER-1

INTRODUCTION

1.1 MOTIVATION

In response to the need for more hygienic, accessible, and intuitive computer interactions, our proposed virtual keyboard and mouse application leverages Convolutional Neural Networks (CNN) for touchless control through hand gestures. Traditional input devices like keyboards and mice pose limitations, particularly in shared or public environments where hygiene is crucial, and for individuals with physical disabilities. By eliminating the need for physical contact, our application reduces the risk of transmitting germs, enhances accessibility, and offers a more natural and intuitive user experience.

1.2 PROBLEM STATEMENT

Interacting with computers using traditional input devices like keyboards and mice can be limiting and inconvenient, particularly in situations where hygiene, accessibility, or ease of use is a concern. As technology advances, there is a growing demand for innovative solutions that offer touchless, intuitive, and efficient ways to control computers.

With the increasing popularity of gesture-based controls and the capabilities of modern computer vision techniques, there is an opportunity to develop a virtual keyboard and mouse system using Convolutional Neural Networks (CNN) that addresses these needs.

Current input devices require physical contact, which can be problematic in shared or public environments where hygiene is critical. Additionally, individuals with physical disabilities may find traditional input methods challenging to use. Existing gesture recognition systems often lack accuracy, real-time performance, or user-friendly interfaces, limiting their practicality and adoption

1.3 PROJECT OBJECTIVE

The primary objective of this project is to develop an innovative virtual keyboard and mouse system that employs Convolutional Neural Networks (CNN) to recognize and interpret hand gestures in real-time. By harnessing the power of CNNs, the system aims to provide a touchless, efficient, and user-friendly alternative to traditional input devices. This approach not only enhances hygiene by eliminating the need for physical contact but also makes computer interactions more accessible for individuals with physical disabilities. The system should accurately detect various hand gestures and translate them into corresponding keyboard and mouse actions, such as typing, clicking, dragging, and scrolling.

In addition to ensuring high accuracy and responsiveness in gesture recognition, the project aims to deliver a seamless user experience with minimal latency. The application should process video input from a camera in real-time, providing immediate feedback and smooth operation. A virtual keyboard interface will be developed to map recognized gestures to specific keys, with visual feedback to improve user interaction. Similarly, the virtual mouse functionality will enable precise cursor movement and control. The user interface will be designed to be intuitive and easy to set up, guiding users through calibration and gesture performance.

Furthermore, the project seeks to create a robust and versatile application that can operate effectively under various lighting conditions and backgrounds. The system should be optimized for high performance and tested extensively to ensure reliability and effectiveness. By packaging the solution into an easily deployable application compatible with common operating systems and standard webcam hardware, the project aims to make this advanced touchless interaction technology widely accessible. The ultimate goal is to revolutionize the way users interact with their computers, providing a more hygienic, accessible, and efficient alternative to traditional input devices.

CHAPTER 2 LITERATURE SURVEY

Cursor control applications that use hand gestures are used in a variety of ways, however, the majority of the time, wearing a DataGlove is required. This reduces the efficiency of performance between the user and the system. The intricacy of the system is also a concern in this procedure. There are two types of gesture recognition for HCI: hardware-based and computer vision-based. Quam (1990) offered one of the first hardware-based systems in which the user had to wear a bulky DataGlove to utilize the system. Though this method provides high accuracy control, it is difficult to use because some gestures are not appropriate for everyone and are also impractical for mass users in the real world.

Abhik Banerjee and Abhirup Ghosh proposed a work titled "Mouse Control Using a Web Camera Based on Colour Detection" in 2014, where the technology is described. Hand motions were recorded with a camera that used a colour detection technique. Their work is limited by the fact that the operational background must be light and no brightly coloured things are present. It performs nicely on certain highend PCs. Based on directly extracting fingers from salient hand edges, Yimin Zhou, Guolai Jiang, and Yaorong Lin published "A novel finger and hand pose estimation technique for real-time hand gesture recognition" in 2016. Hand posture is segmented and defined based on finger positions, palm centre location, and wrist position, taking into account hand geometrical qualities.

In 2017, Aashni Haria, Archanasri Subramanian, Nivedhitha Asokkumar, Shristi Poddar, Jyothi S Nayak developed "Hand Gesture Recognition for Human-Computer Interaction" [9] based on background extraction and contours detection system. But it is very slow to work with.

In 2018, Abhilash SS, Lisho Thomas, Naveen Wilson, Chaithanya C published a paper on "VirtualMouse Using Hand Gesture" [10] which designed to work with the colour detection system, functions work on the number of colours detected. But it can perform only a few mouse actions, and does not work without static background

2.1 Existing System

1. **2.1.1 Optical mouse and keyboard:** Uses a light-emitting diode and photodiodes to detect movement relative to the underlying surface. Digital image correlation, a technology pioneered by the defense industry for tracking military targets. Use image sensors to image naturally occurring texture in materials such as wood, cloth, mouse pads and Image captures in continuous succession and comparison to determine mouse movement.

Limitations

1. Special hardware required.
2. Again, specific surface requirements.

2.1.2 Mechanical Mouse and keyboard : A single ball that could rotate in any direction. As part of the hardware package of the Xerox Alto computer. Detection of the motion of the ball was light based with the help of chopper wheels.

Limitations

1. Cannot provide high precision performance.
2. Has specific surface requirements to operate.
3. Needs more desk space when compared with a trackball.

1.1.2 TrackBall : The user rolls the ball with the thumb, fingers, or the palm of the hand to move a cursor. Large tracker balls are common on CAD workstations for easy precision. Before the advent of the touchpad, small trackballs were common on portable computers.

Limitations

1. Usually not as accurate as a mouse.
2. Ball mechanism of trackballs requires more frequent cleaning than a mouse.
3. Not very user friendly

CHAPTER-3

SOFTWARE REQUIREMENT SPECIFICATION

This chapter gives an overview of the software and hardware components required for our project.

3.1 SOFTWARE REQUIREMENTS

Operating System	:	Windows 8
Frame work/Librarires	:	Mediapipe, OpenCV
Coding Language	:	Python 3.7

3.2 HARDWARE REQUIREMENTS

System	:	intel i5 or above
Storage	:	Sufficient storage
Processor	:	Modern multi-core processor for real-time video processing and CNN computations.

3.3 FUNCTIONAL REQUIREMENTS

These are the requirements that the end user specifically demands as basic facilities that the system should offer. All these functionalities need to be necessarily incorporated into the system as a part of the contract. These are represented or stated in the form of input to be given to the system, the operation performed and the output expected.

- Hand Detection and Tracking
- Gesture Recognition
- Mouse Control
- Keyboard Control
- Calibration User Interface

- User Interface
- Settings and Customization

3.4 NON-FUNCTIONAL REQUIREMENTS

These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to other. They are also called non- behavioural requirements.

They basically deal with issues like:

- Portability
- Security
- Maintainability
- Reliability
- Scalability
- Performance
- Reusability
- Flexibility

CHAPTER-4 SYSTEM DESIGN

4.1 System Design

In this phase, the system and software design documents are prepared based on the requirement specification document. This helps define the overall system architecture for creating a virtual mouse and keyboard using Convolutional Neural Networks (CNN).

There are two kinds of design documents developed in this phase:

High-Level Design (HLD)

- Brief description and name of each module
- An outline about the functionality of every module
- Interface relationship and dependencies between modules
- Database tables identified along with their key elements
- Complete architecture diagrams along with technology details

Low-Level Design(LLD)

- Functional logic of the modules
- Database tables, which include type and size
- Complete detail of the interface
- Addresses all types of dependency issues
- Listing of error messages

4.2 SYSTEM ARCHITECTURE:

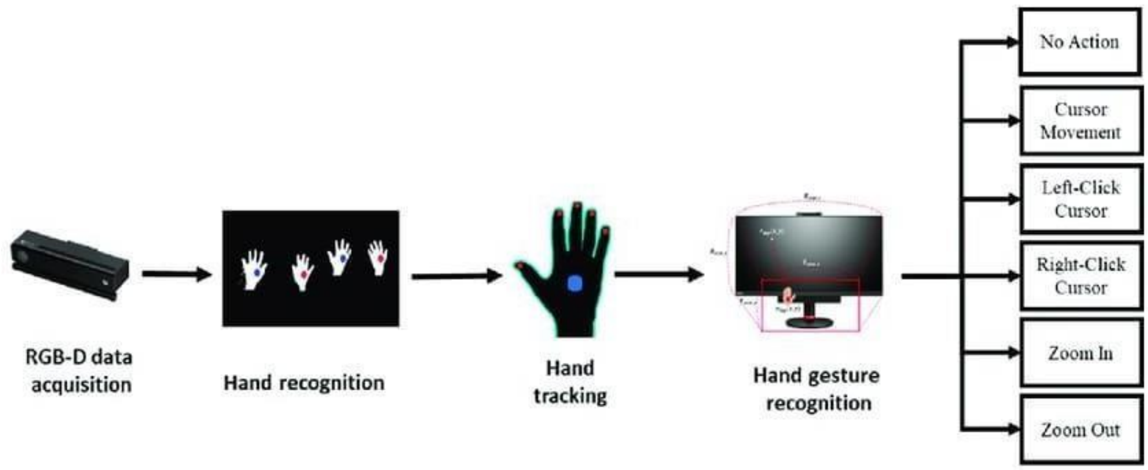


Figure-4.2.1 System Architecture

4.2 UML Design:

Unified Modeling Language (UML) is a general purpose modeling language. The main aim of UML is to define a standard way to visualize the way a system has been designed. It is quite similar to blueprints used in other fields of engineering.

UML is not a programming language; it is rather a visual language. We use UML diagrams to portray the behaviour and structure of a system, UML helps software engineers, businessmen and system architects with modeling, design and analysis. The Object Management Group (OMG) adopted Unified Modeling Language as a standard in 1997. It's been managed by OMG ever since. International Organization for Standardization (ISO) published UML as an approved standard in 2005. UML has been revised over the years and is reviewed periodicall

Do we really need UML?

- Complex applications need collaboration and planning from multiple teams and hence require a clear and concise way to communicate amongst them.
- Businessmen do not understand code. So UML becomes essential to communicate with non programmer's essential requirements, functionalities and processes of the system.

- A lot of time is saved down the line when teams are able to visualize processes, user interactions and static structure of the system.
- UML is linked with object oriented design and analysis. UML makes the use of elements and forms associations between them to form diagrams. Diagrams in UML can be broadly classified as:

The Primary goals in the design of the UML are as follows:

- Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
- Provide extendibility and specialization mechanisms to extend the core concepts.
- Be independent of particular programming languages and development process.
- Provide a formal basis for understanding the modeling language.
- Encourage the growth of OO tools market.
- Support higher level development concepts such as collaborations, frameworks, patterns and components.
- Integrate best practices.

Types of UML Diagrams:

Structural Diagrams:

Capture static aspects or structure of a system. Structural Diagrams include: Component Diagrams, Object Diagrams, Class Diagrams and Deployment Diagrams.

Behaviour Diagrams:

Capture dynamic aspects or behaviour of the system. Behaviour diagrams include: Use Case Diagrams, State Diagrams, Activity Diagrams and Interaction Diagrams.

The image below shows the hierarchy of diagrams according to UML

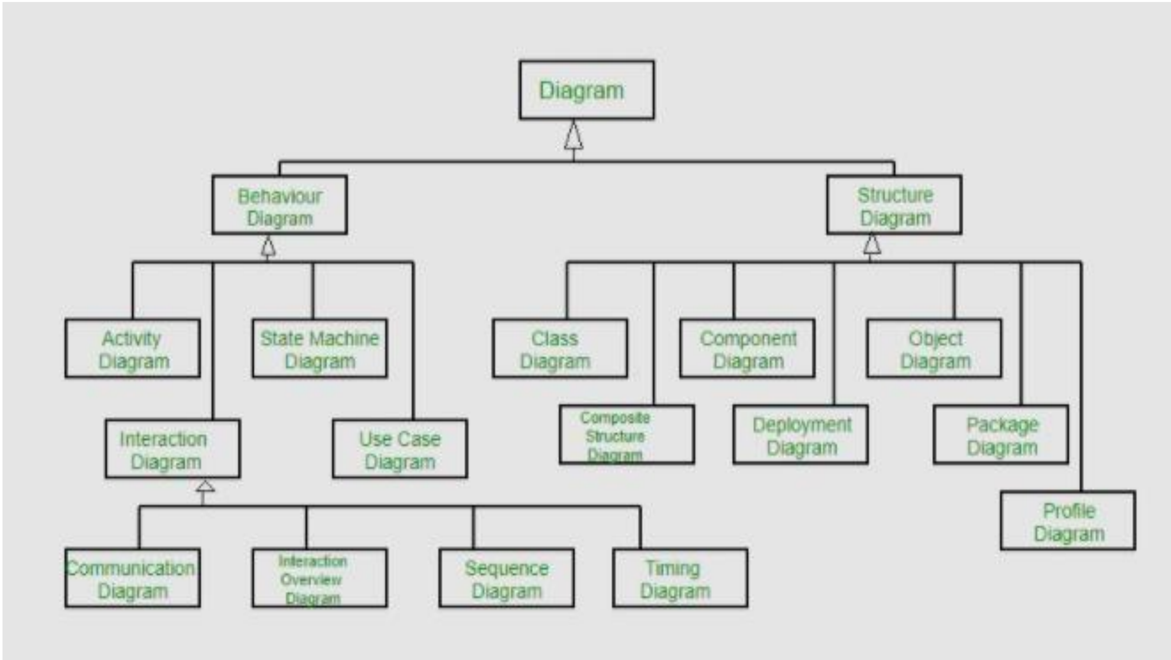


Figure-4.2.1 UML Hierarchy diagrams

4.3.1 CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

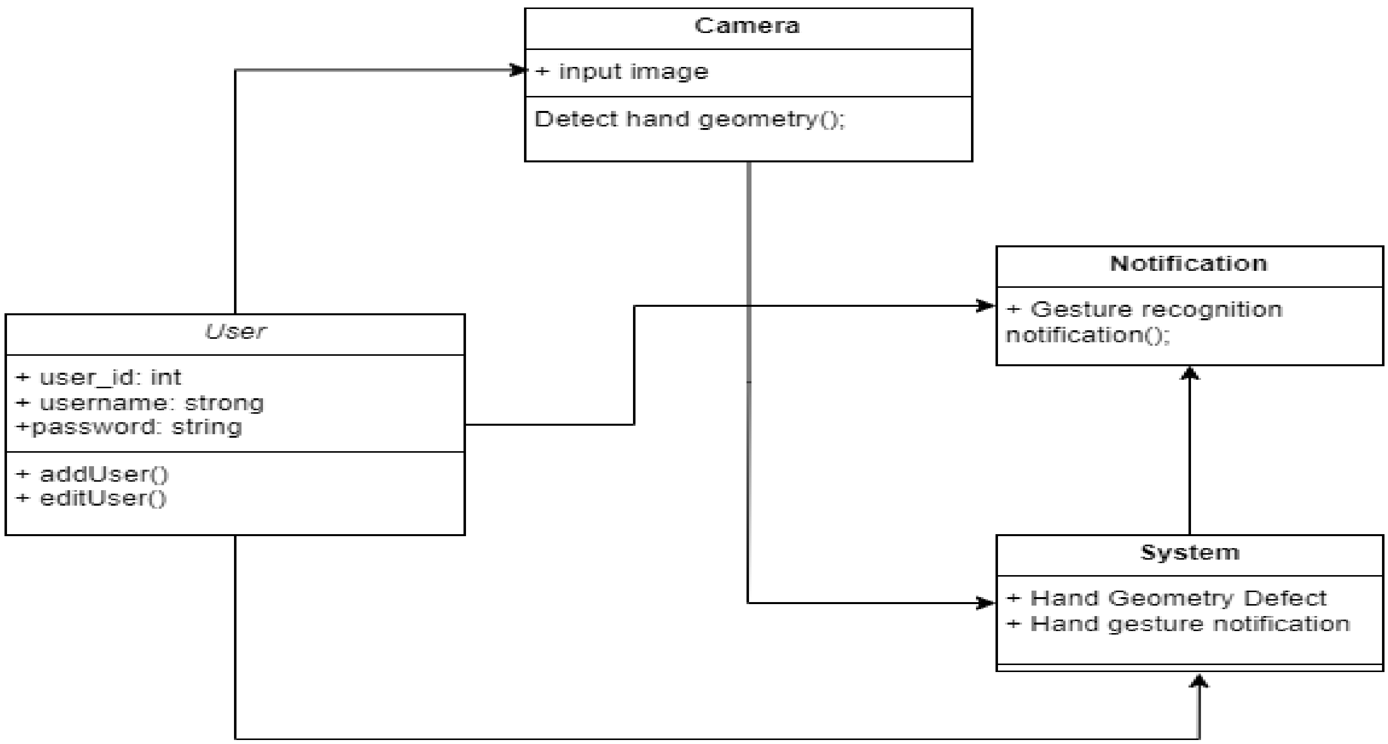


Figure-4.3.1.1 Class Diagram

4.3.2 USE CASE DIAGRAM:

A use case diagram in the Unified Modelling Language (UML) is a type of behavioural diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted

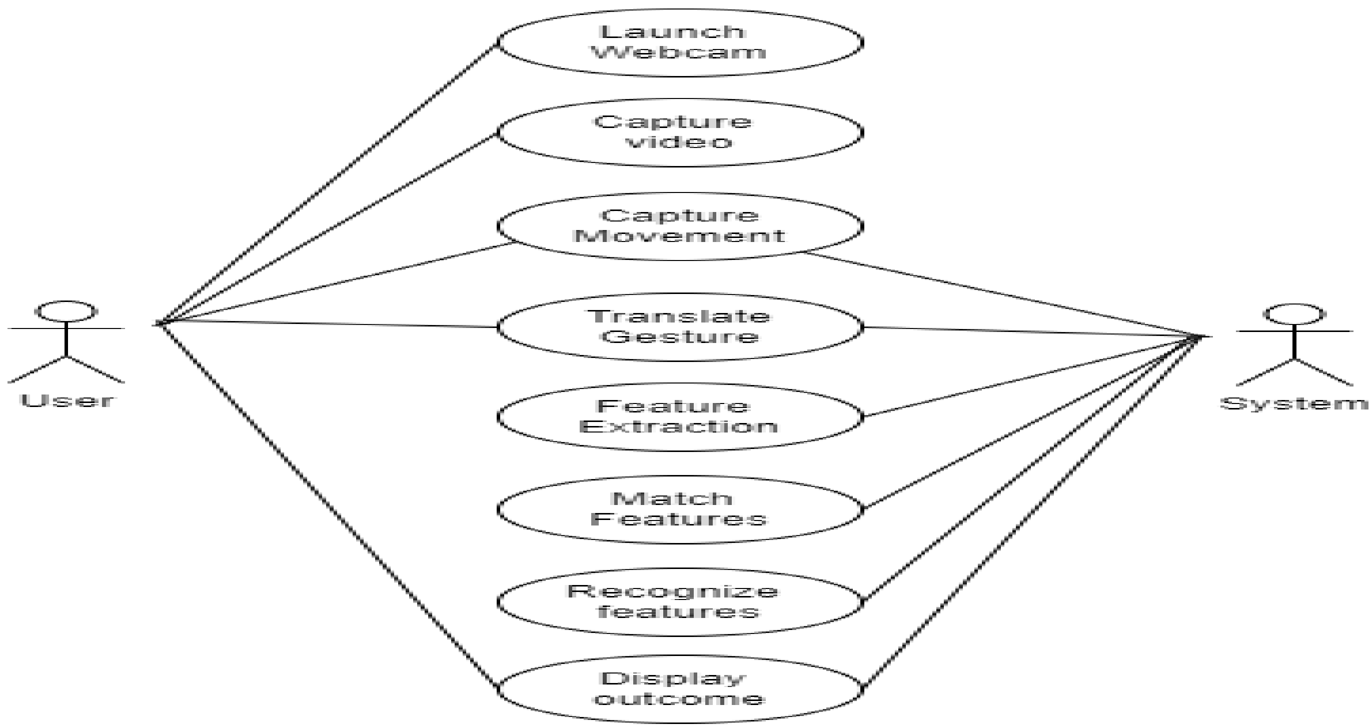


Figure-4.3.2.1 Use Case Diagram.

43.3 COMPONENT DIAGRAM:

A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical components in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required functions is covered by planned development.

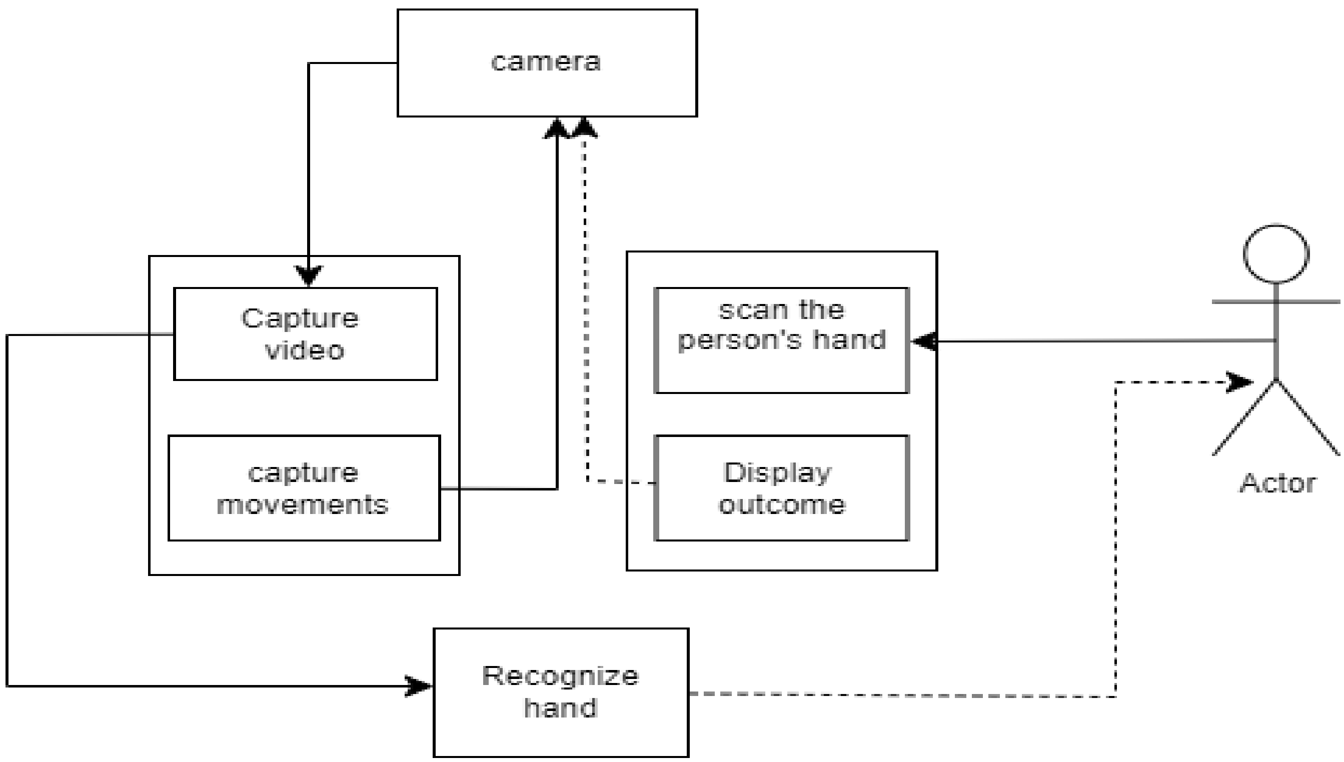


Figure-4.3.3.1 Component Diagram

4.3.4 SEQUENCE DIAGRAM:

A sequence diagram in Unified Modelling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

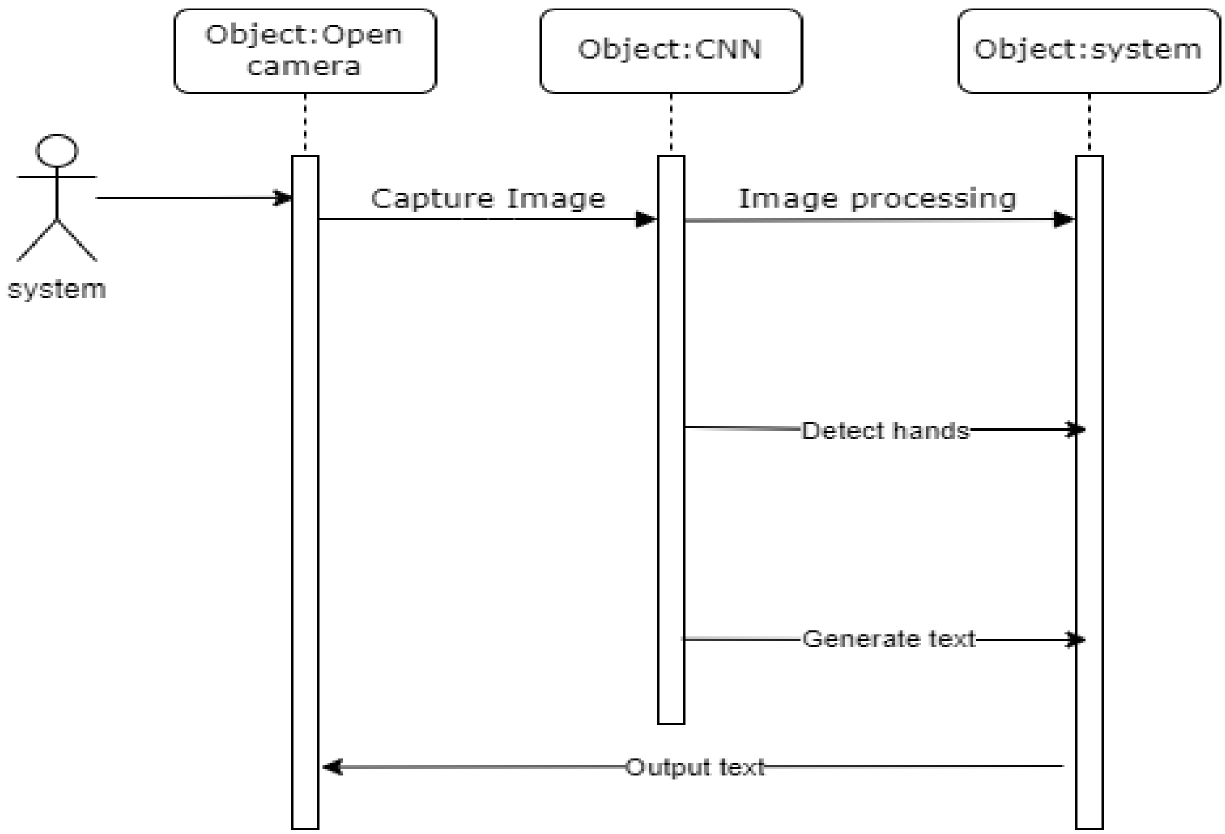


Figure-4.3.4.1 Sequence Diagram

4.3.5 ACTIVITY DIAGRAM:

In UML, an activity diagram is used to display the sequence of activities. Activity diagrams show the workflow from a start point to the finish point detailing the many decision paths that exist in the progression of events contained in the activity.

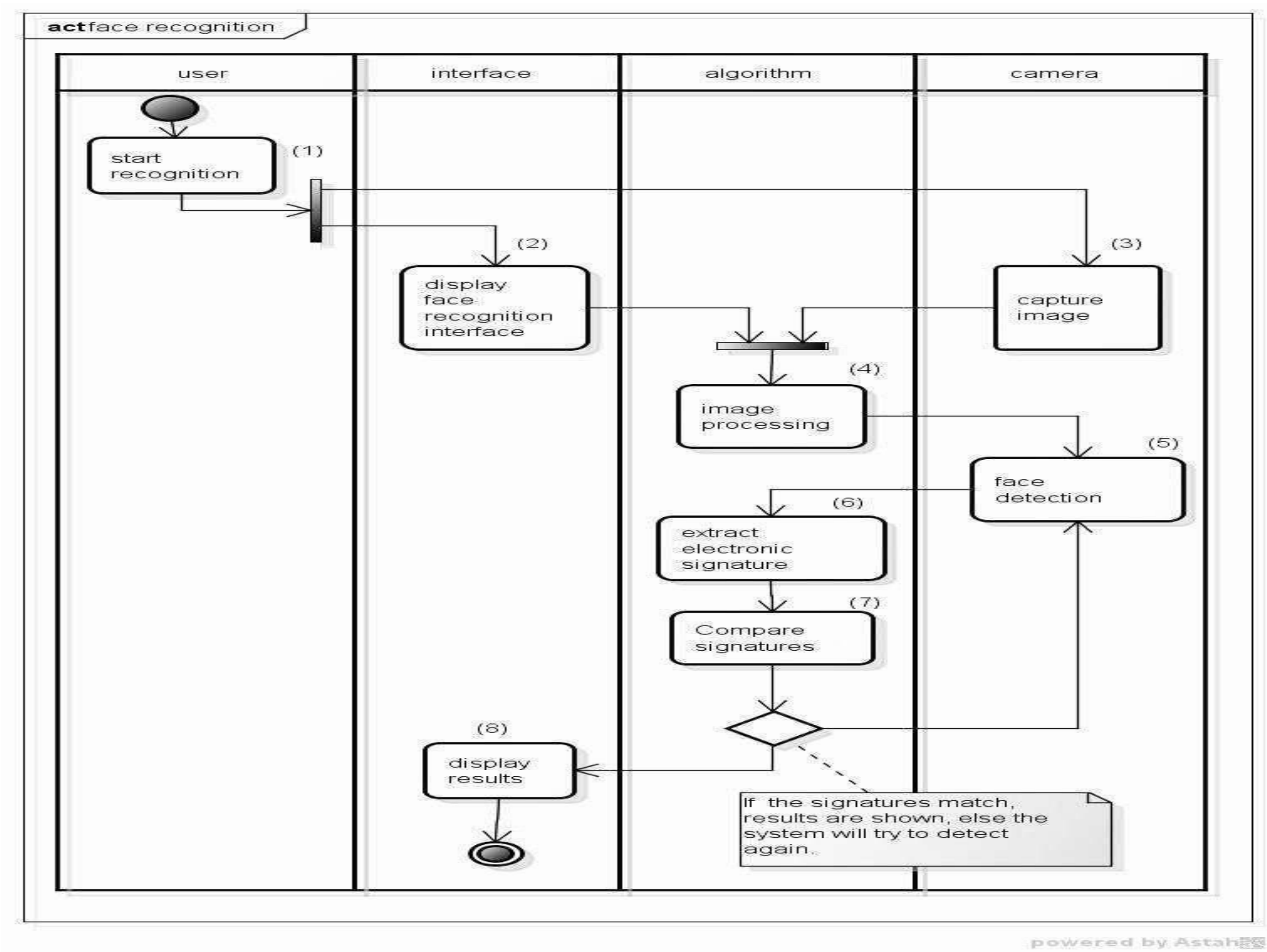


Figure-4.3.5.1 Activity diagram

4.3 TECHNOLOGY DESCRIPTION

Convolutional Neural Networks (CNN):

Convolutional Neural Networks (CNN) are a class of deep learning algorithms particularly effective in image recognition and computer vision tasks. They will be employed to develop the core gesture recognition functionality of the virtual keyboard and mouse system. CNNs will process video input from a camera to detect and classify hand gestures in real-time. These networks are composed of layers that automatically and adaptively learn spatial hierarchies of features, enabling precise and efficient gesture recognition.

Real-Time Video Processing:

Real-time video processing is crucial for a responsive and seamless user experience. The system will capture video input using a standard webcam and process it in real-time to detect hand gestures. Techniques such as frame extraction, preprocessing (e.g., normalization, resizing), and feature extraction will be applied to each frame before feeding it into the CNN for gesture recognition. Ensuring low latency in processing is essential to provide immediate feedback and smooth operation.

User Interface (UI):

The user interface will be designed to be intuitive and user-friendly, providing clear visual feedback and guidance for performing gestures. The UI will include a virtual keyboard that appears on the screen, where recognized gestures map to specific keys, and visual cues indicate "key presses." Similarly, the virtual mouse interface will allow users to move the cursor, click, drag, and scroll through hand gestures. The UI design will focus on usability, ensuring that users can easily set up, calibrate, and use the system.

Python:

Python will be the primary programming language used to develop the backend of the application, integrating the CNN model and handling gesture recognition logic. Python's extensive libraries and frameworks for machine learning and computer vision, such as TensorFlow and OpenCV, will facilitate the development and deployment of the CNN model. Additionally, Python will manage the real-time video processing and interaction between the user interface and the gesture recognition system.

TensorFlow and OpenCV:

TensorFlow is a powerful machine learning framework that will be used to build, train, and deploy the CNN model for gesture recognition. OpenCV (Open Source Computer Vision Library) will assist with video capture, image processing, and feature extraction tasks. Together, these tools will enable the efficient development of a robust gesture recognition system capable of real-time performance.

WebSocket:

WebSocket is a communication protocol that provides full-duplex communication channels over a single TCP connection. It will be used to facilitate real-time communication between the frontend and backend of the application. WebSocket ensures that recognized gestures are promptly communicated to the user interface, allowing for instantaneous updates and interactions.

CHAPTER 5

5.1 IMPLEMENTATION

In this chapter, detailed description of algorithm, flowchart of framework of system is given.

5.1 ALGORITHM

5.1.1 DEEP LEARNING

- Deep Learning is a subset of machine learning. It is basically learning and improving on its own by examining another algorithm
- It works on an artificial neural network that was designed to imitate human thinking and learn capabilities

5.1.2 MEDIAPIPE

- MediaPipe to recognize the hand and the hand key points.
- MediaPipe returns a total of 21 key points for each detected hand.

5.1.3 CLASSIFICATION MODEL CNN (CONVOLUTIONAL NEURAL NETWORK)

A convolution neural network comprises numerous hidden layers that enable in the extraction of information from a picture. CNN has five major layers:

1. Input layer
2. Convolution layer
3. ReLU layer
4. Pooling layer
5. Fully connected layer

1. INPUT LAYER:

- This layer is given as the input for the CNN model. The input is fed as the matrix which contains the pixel values in it. • In this layer the input image is fed with the three dimensions height , width and depth.

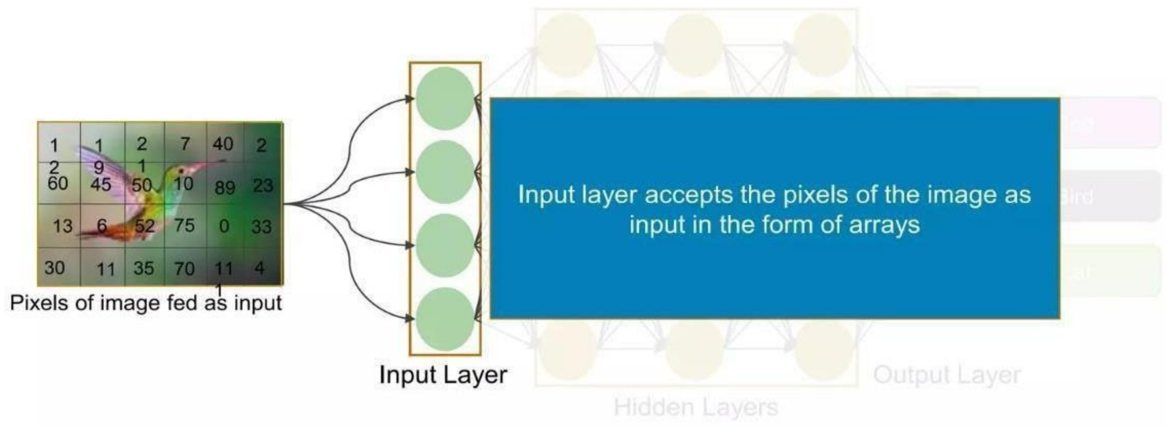


Fig.3: Image to demonstrate how input is fed to CNN

2. CONVOLUTION LAYER:

- In this layer, we perform the convolution operation. It is the main layer in this model as we extract various features from the provided image.
- In this layer, every image is shown as pixel values. Various filters are used in this layer to perform the convolution operation.
- First we take the image with 5 x 5 dimensions which consists of pixel values of either 0 or 1. Then we take a filter matrix with 3x3 dimensions to perform the convolution operation.
- We slide the filter matrix of 3 x 3 dimensions over the image with 5 x 5 dimensions to get the convolved feature map.
- We calculate the dot product of the filter matrix and the pixel values of an image.

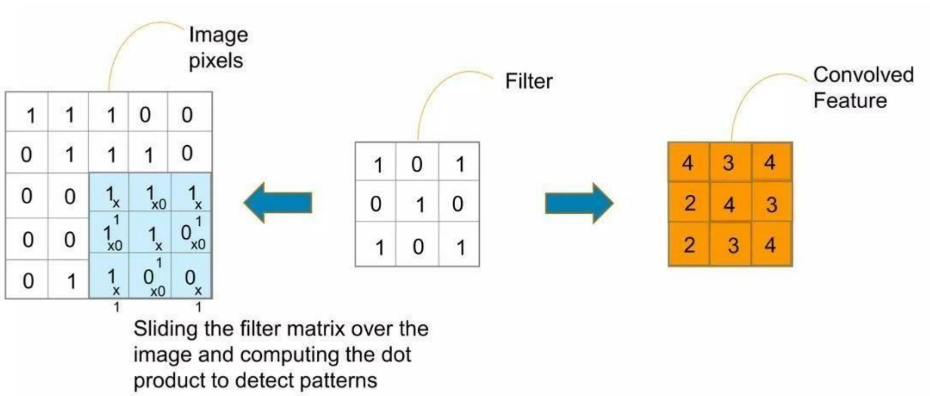


Fig.4: Image to demonstrate how convolution operation takes place

3. ReLU (RECTIFICATION LINEAR UNITS) LAYER:

- In this layer we perform element-wise operations. We set all the negative values in the pixel values as 0. • By performing the above operation, the non-linearity is added to the network.
- In this layer, multiple convolutions are performed on the image, and the image is scanned with ReLU layer to locate the various features in the image. • After performing all these, we get the rectified feature map.

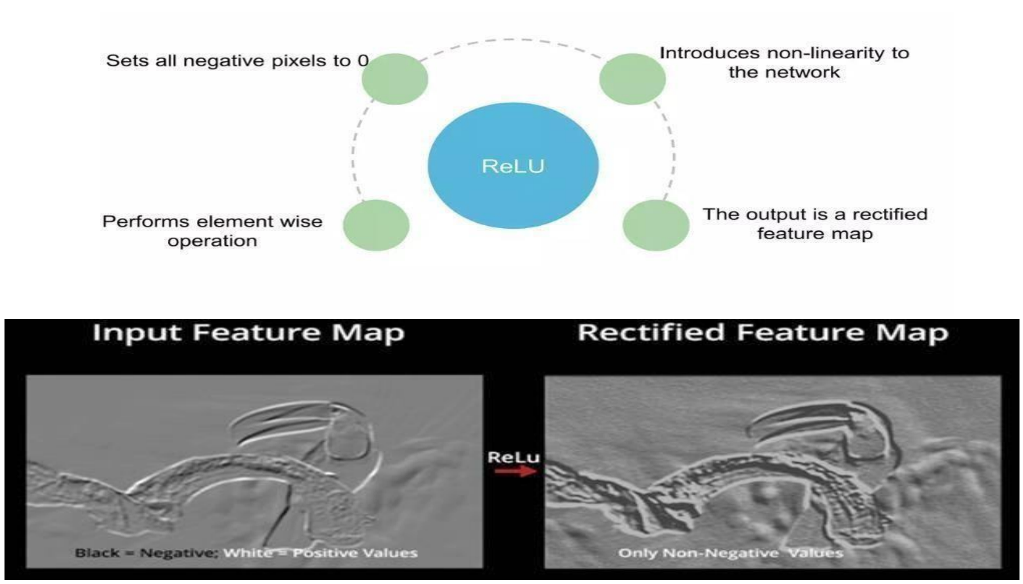


Fig.5: After ReLU operation , Rectified Feature Map is obtained

4. POOLING LAYER:

- It is a sub-sampling operation that decreases the dimensions of the feature map. The rectified feature map which is obtained in the above layer is moved through a pooling layer to obtain a pooled feature map.

- This layer includes the recognition of various elements in the image like the corners, body, eyes, feathers and beak.

- The pooling layer includes various filters to recognize various image elements like edges, corners, body, feathers, eyes, and beak.

- There are two types of pooling.They are Average Pooling and Max Pooling. We generally use Max Pooling as it produces more accurate results than the average pooling.

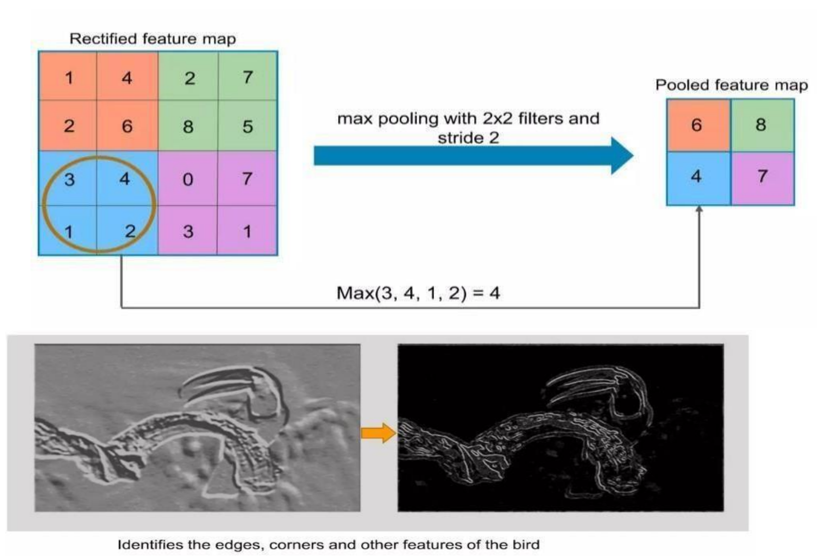


Fig.6: After pooling operation, Pooled Feature Map is obtained

- Flattening is the next process in pooling process.

- Flattening is the process of transforming all of the pooled feature map’s resultant 2D arrays into a one long vector.

Fig.7: Flattening

5. FULLY CONNECTED LAYER

- For the categorization and classification, the flattened matrix is fed as input to this layer.
- In this layer, it retains the results of the labels which are computed for classification and gives a class to the image, and assigns an exact label for it.

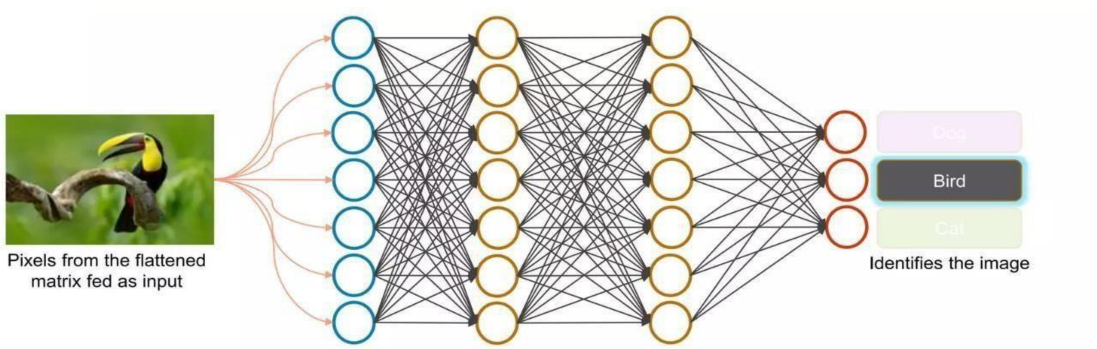


Fig.8: Fully Connected Layer

In this chapter, implementation of algorithm is described. Results will be explained in terms of tables and graphs etc.

5.2 Experimental Work

5.2.1 IMPLEMENTATION

LOAD THE DATASETS FOR THE NUMBERS AND ALPHABETS

Fig 9 : Dataset images for numbers

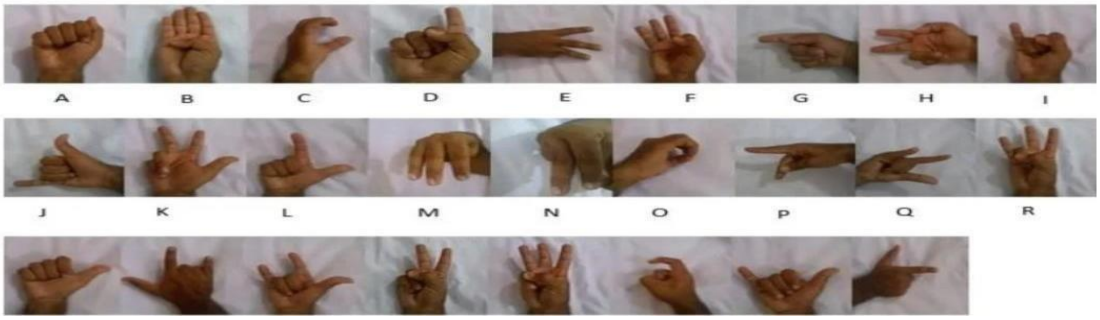


Fig 10: Dataset images for alphabets

5.4.1 Code for mouse control

```
import  cv2  as  cv  import  numpy  as  np  from
pynput.mouse    import    Button,Controller
import wx import imutils
 mouse = Controller()
app = wx.App(False)
(sx,sy)  =  wx.GetDisplaySize()
(camx,camy)  =  (320,240)  cam  =
cv.VideoCapture(0)
cam.set(3,camx)
cam.set(4,camy)    mlocold    =
np.array([0,0])   mouseloc   =
np.array([0,0])  damfac  =  2.5
pinch_flag = 0 while(1):
    ret,img  =  cam.read()  img  =  cv.GaussianBlur(img,(5,5),0)  hsv_img  =
    cv.cvtColor(img,cv.COLOR_BGR2HSV)                  mask               =
    cv.inRange(hsv_img,np.array([65,60,60]),np.array([80,255,255]))
    mask_open      =      cv.morphologyEx(mask,cv.MORPH_OPEN,np.ones((5,5)))
    mask_close  =  cv.morphologyEx(mask_open,cv.MORPH_CLOSE,np.ones((20,20)))
    mask_final = mask_close conts,_ =
cv.findContours(mask_final.copy(),cv.RETR_EXTERNAL,cv.CHAIN_APPROX_SIMPLE)
    cv.drawContours(img,conts,-1,(255,0,0),3) if(len(conts)==2):
        if(pinch_flag==1):
            pinch_flag = 0 mouse.release(Button.left)
        x1,y1,w1,h1  =  cv.boundingRect(conts[0])  x2,y2,w2,h2
        = cv.boundingRect(conts[1])
        #cv.rectangle(img,(x1,y1),(x1+w1,y1+h1),(255,0,0),2)
```

```
        #cv.rectangle(img,(x2,y2),(x2+w2,y2+h2),(255,0,0),2
        ) cx1 = round(x1+w1/2) cy1 = round(y1+h1/2) cx2 =
        round(x2+w2/2)      cy2      =      round(y2+h2/2)
        cv.line(img,(cx1,cy1),(cx2,cy2),(255,0,0),2)  cx  =
        round(cx1/2+cx2/2)    cy    =    round(cy1/2+cy2/2)
        cv.circle(img,(cx,cy),2,(0,0,255),2)   mouseloc   =
        mlocold+((cx,cy)-mlocold)/damfac  mouse.position  =
        (round(sx -
(mouseloc[0]*sx/camx)),round((mouseloc[1]*sy/camy)))
        mlocold = mouseloc
 elif(len(conts)==1):
        if(pinch_flag==0):
            pinch_flag = 1 mouse.press(Button.left)
 x,y,w,h    =    cv.boundingRect(conts[0])    cx    =
        round(x+w/2)      cy      =      round(y+h/2)
        cv.circle(img,(cx,cy),20,(0,0,255),2)
        mouseloc = mlocold+((cx,cy)-mlocold)/damfac
        mouse.position = (round(sx -
mouseloc[0]*sx/camx),round(mouseloc[1]*sy/camy))
        mlocold = mouseloc
 cv.imshow("cam",img)
    cv.waitKey(10)
```

5.4.1 Code for keyboard control :

```
import cv2 as cv import numpy as np import
imutils import json import pyautogui import
time cam = cv.VideoCapture(0) arr=[]
nums=["1","2","3","4","5","6","7","8","9","0"]
row1=["Q","W","E","R","T","Y","U","I","O","P"]
row2=["A","S","D","F","G","H","J","K","L"]
row3=["Z","X","C","V","B","N","M"]
row4=["space","enter","backspace","shift"]
row5=["left","up","down","right"]
row6=["volumeup","volumedown","volumemute"]
x=10 y=20 for i in range(0,10):
    data={} data["x"]=x
    data["y"]=y
    data["w"]=100
    data["h"]=80
    data["value"]=nums[i]
    arr.append(data)
    x=x+100
y=100 x=10 for i in
range(0,10):
    data={} data["x"]=x
    data["y"]=y
    data["w"]=100
    data["h"]=80
```

```
    x=510 y=340
    cv.rectangle(img,(x,y),(x+250,y+80),(0,255,255),2)
    cv.putText(img,row4[2],(x+70,y+40),cv.FONT_HERSHEY_SIMPLEX,1,(0,0,255),2,cv.LI
NE_AA,False) x=760 y=340
    cv.rectangle(img,(x,y),(x+200,y+80),(0,255,255),2)
    cv.putText(img,row4[3],(x+70,y+40),cv.FONT_HERSHEY_SIMPLEX,1,(0,0,255),2,cv.LI
NE_AA,False) x=110 y=420
    cv.rectangle(img,(x,y),(x+200,y+80),(0,255,255),2)
    cv.putText(img,row5[0],(x+100,y+40),cv.FONT_HERSHEY_SIMPLEX,1,(0,0,255),2,cv.L
INE_AA,False) x=310 y=420
    cv.rectangle(img,(x,y),(x+200,y+80),(0,255,255),2)
    cv.putText(img,row5[1],(x+100,y+40),cv.FONT_HERSHEY_SIMPLEX,1,(0,0,255),2,cv.L
INE_AA,False) x=510 y=420
    cv.rectangle(img,(x,y),(x+200,y+80),(0,255,255),2)
    cv.putText(img,row5[2],(x+100,y+40),cv.FONT_HERSHEY_SIMPLEX,1,(0,0,255),2,cv.L
INE_AA,False) x=710 y=420
    cv.rectangle(img,(x,y),(x+200,y+80),(0,255,255),2)
    cv.putText(img,row5[3],(x+100,y+40),cv.FONT_HERSHEY_SIMPLEX,1,(0,0,255),2,cv.L
INE_AA,False)
    x=10 y=500
    cv.rectangle(img,(x,y),(x+200,y+80),(0,255,255),2)
    cv.putText(img,"V+",(x+60,y+40),cv.FONT_HERSHEY_SIMPLEX,1,(0,0,255),2,cv.LINE_
AA,False) x=210 y=500
    cv.rectangle(img,(x,y),(x+200,y+80),(0,255,255),2)
```

```
    cv.putText(img,"V-
",(x+60,y+40),cv.FONT_HERSHEY_SIMPLEX,1,(0,0,255),2,cv.LINE_AA,False) x=410
y=500 cv.rectangle(img,(x,y),(x+200,y+80),(0,255,255),2)
cv.putText(img,"Vmute",(x+60,y+40),cv.FONT_HERSHEY_SIMPLEX,1,(0,0,255),2
,cv.LINE_AA,False)
 hsv_img       =       cv.cvtColor(img,cv.COLOR_BGR2HSV)       mask       =
    cv.inRange(hsv_img,np.array([65,60,60]),np.array([80,255,255]))
    mask_open      =      cv.morphologyEx(mask,cv.MORPH_OPEN,np.ones((5,5)))
    mask_close = cv.morphologyEx(mask_open,cv.MORPH_CLOSE,np.ones((20,20)))
    mask_final = mask_close conts,_ =
cv.findContours(mask_final.copy(),cv.RETR_EXTERNAL,cv.CHAIN_APPROX_SIMPLE)
    #cv.drawContours(img,conts,-
    1,(255,0,0),3) if(len(conts)==1): x,y,w,h
    =    cv.boundingRect(conts[0])    cx    =
    round(x+w/2)     cy     =     round(y+h/2)
    cv.circle(img,(cx,cy),20,(0,0,255),2)
    word="" for i in range(len(json_data)):
            if cx>=int(json_data[i]["x"]) and
cx<=int(json_data[i]["x"])+int(json_data[i]["w"]) and
cy>=int(json_data[i]["y"])                          and
cy<=int(json_data[i]["y"])+int(json_data[i]["h"]):
word=json_data[i]["value"]
        #print(word) pyautogui.press(word)
        #pyautogui.PAUSE=2.5
        #time.sleep(1)
 cv.imshow("cam",img)
    cv.waitKey(10)
```

CHAPTER - 6 TESTING

6.1 TESTING DEFINATION:

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

6.2 Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Since, the grey box testing includes access to internal coding for designing test cases. Grey box testing is performed by a person who knows coding as well as testing.

Outcomes Possible:

Pass: The test case successfully validates the expected behaviour of the application, indicating that specific functionality works as intended.

Fail: The test case fails to validate the expected behaviour, indicating a defect or issue in the application. This outcome requires further investigation and fixing the identified problem.

Error: An error occurs during the test execution due to unexpected system behaviour or exceptions. This could indicate a bug or potential issue that needs to be addressed.

CHAPTER – 7

RESULTS

- After running the program for mouse control we can see the following pages

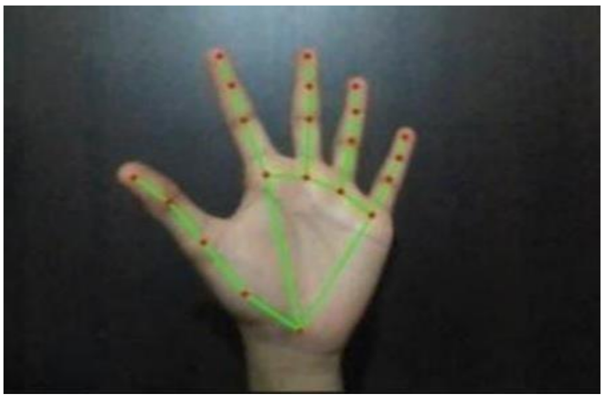


FIG: PERFORMING NEUTRAL GESTURE

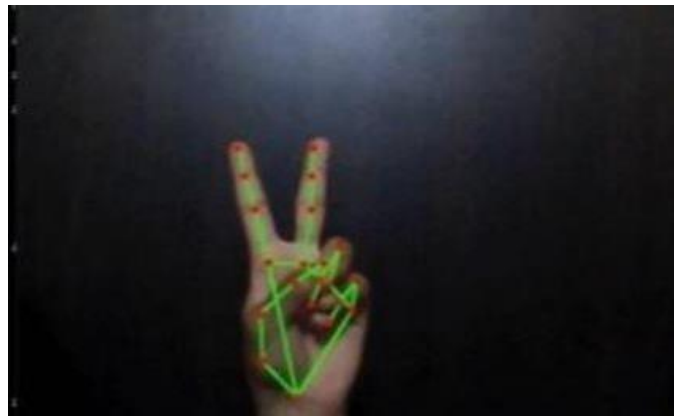
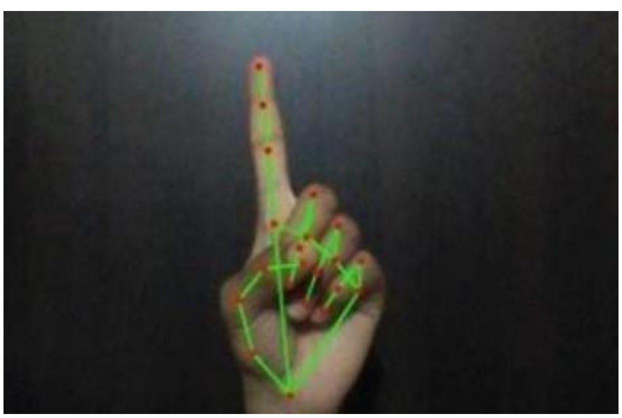


FIG : PERFORMING CURSOR MOVE



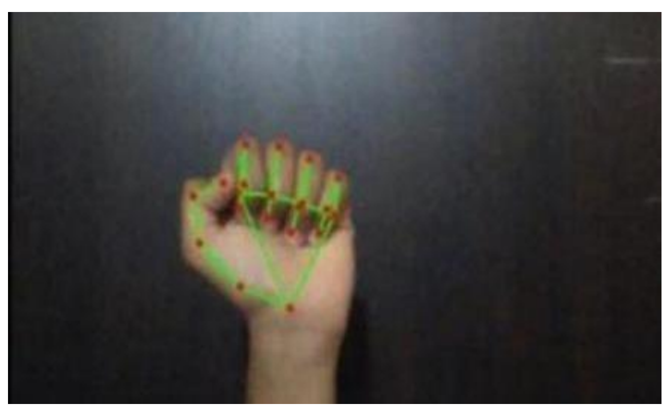


FIG: PERFORMING RIGHT CLICK

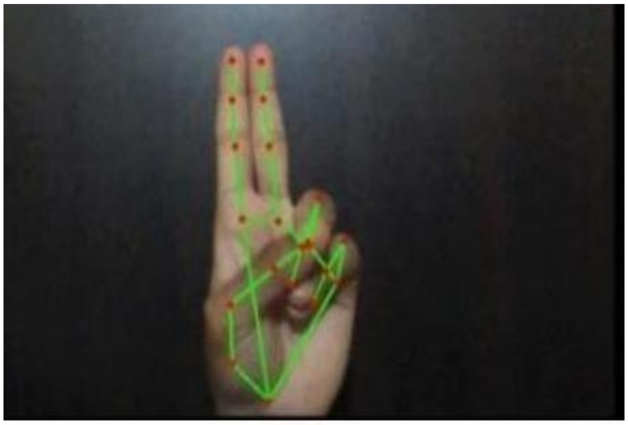


FIG : PERFORMING DRAG AND DROP

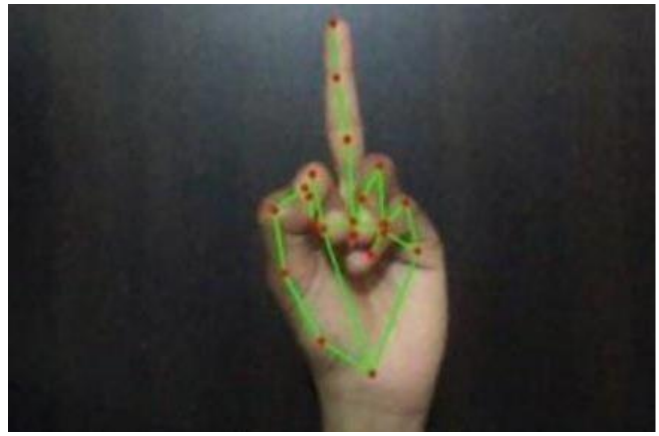
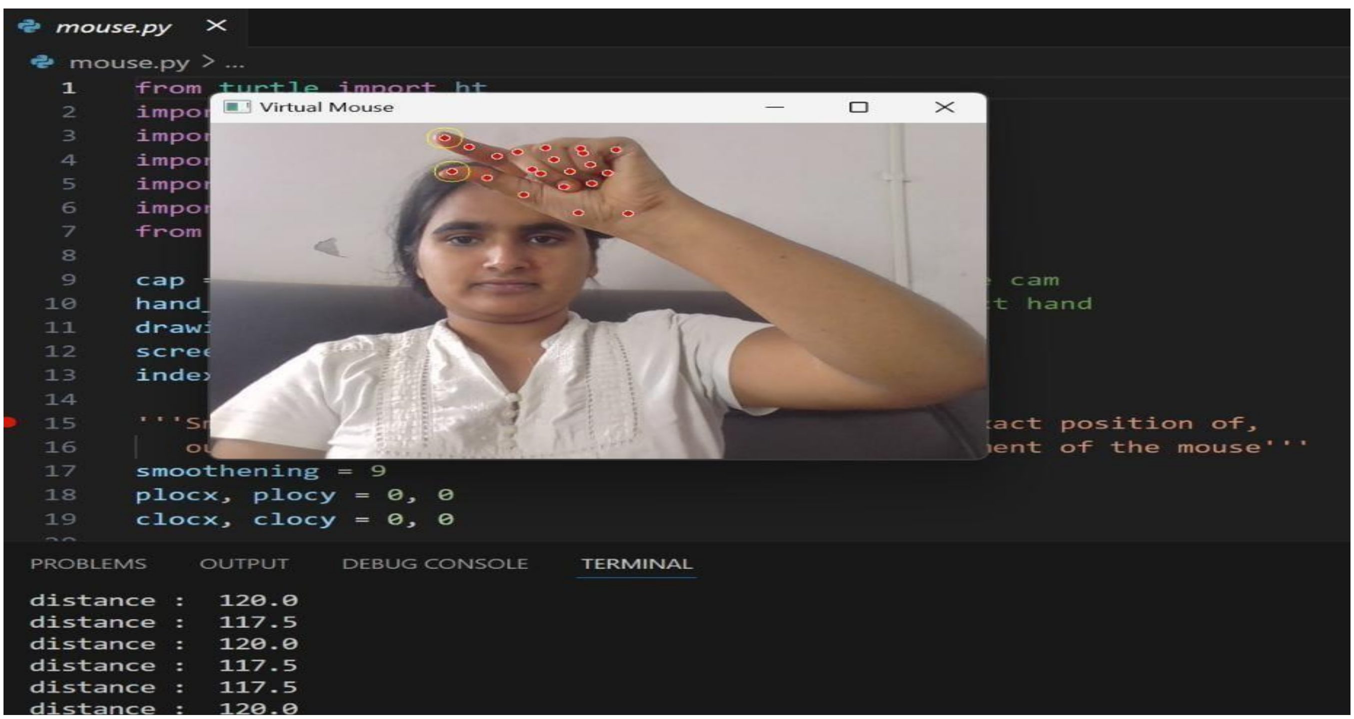


FIG: PERFORMING DOUBLE CLICK

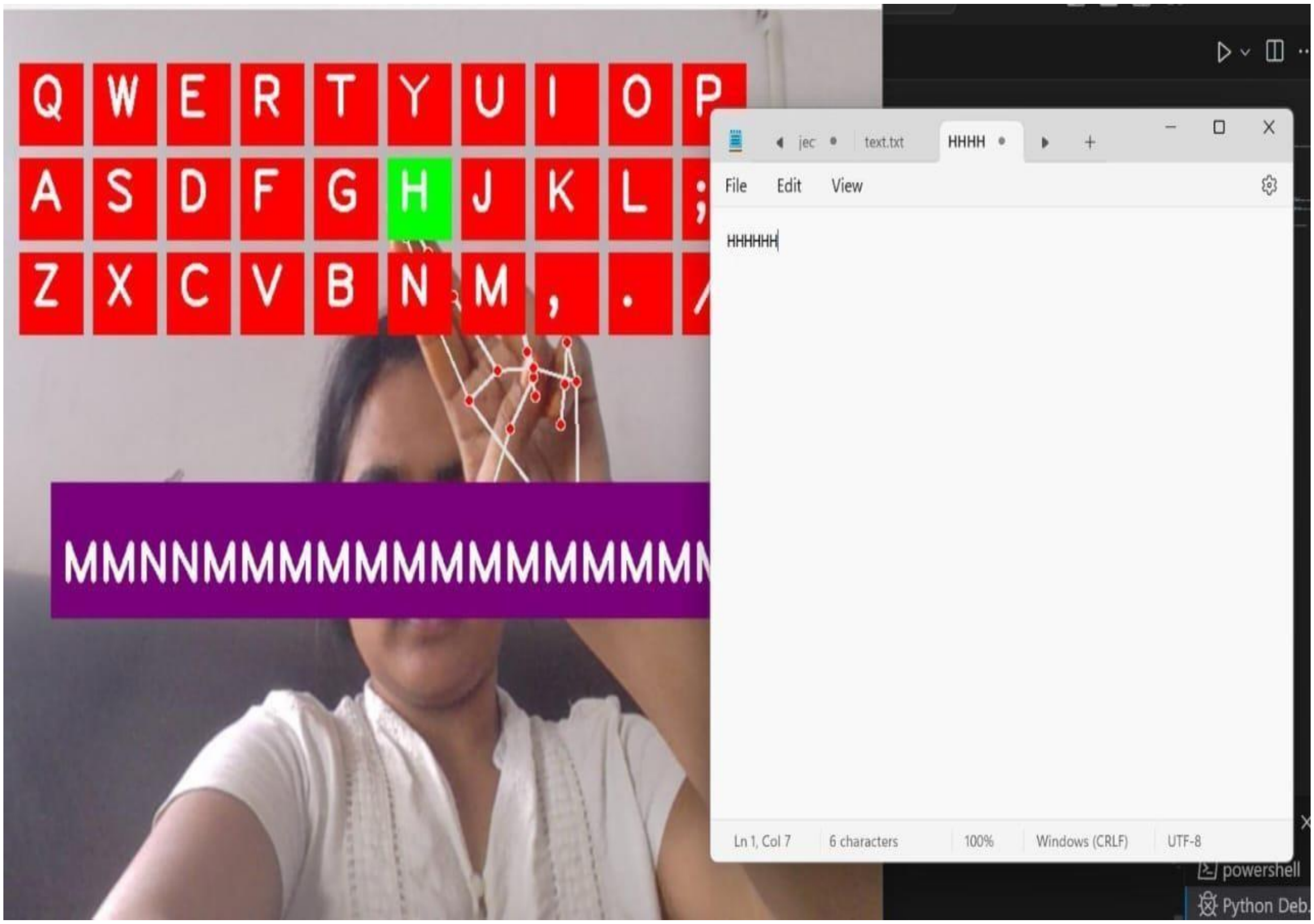
FIG: PERFORMING RIGHT

Final output:



After running the program for keyboard control we can see the following pages:

Final output :



CHAPTER 8

8.1 CONCLUSION

A few strategies have to be devised since precision and efficiency are vital in making the application as helpful as a real mouse and keyboard. There is no longer any need for a real mouse and keyboard to implement such an application. This motion-tracking mouse records every click and drag of a real mouse (virtual mouse).

Improvements and new features are required to make the application more user-friendly, accurate, and adaptable to different contexts. The following is a list of the features and upgrades that are needed:

a) Smart movement: It is necessary to have an adaptive zoom in/out capability that can automatically vary the focus rate dependent on the distance between the user and the camera to cover a greater distance. To get a faster reaction time, hardware factors such as the CPU speed, RAM size, and camera features are important considerations.

b) Improved Accuracy & Performance If the software runs on a high-end computer with an excellent camera, it may operate better under varied lighting conditions.

8.2 FUTURE SCOPE

- There will be no use of hardware like keyboard and mouse.
- Only webcam is required to control keyboard and mouse controls.
- It is more accurate one yet sometimes it is not so fast.
- It is very helpful to control only through webcam.

8.3 REFERENCES

- Quam, D.L., et.al. (1990). Gesture Recognition with a Dataglove. In IEEE conference on Aerospace and Electronics (pp. 755-760).
- Baldauf, M., and Frohlich, p. (2013). Supporting Hand Gesture Manipulation of Projected Content with mobile phones. In the European conference on computer vision (pp. 381-390).
- Mayur, Yeshi., Pradeep, Kale., Bhushan, Yeshi., & Vinod Sonawane. (2016). Hand Gesture Recognition for Human-Computer Interaction. In the international journal of scientific development and research (pp. 9-13).

CHAPTER 9

PLAGIARISM REPORT