"Gesture Recognition using OpenCV"

A Project Report Submitted to
Rajiv Gandhi Proudyogiki Vishwavidyalaya

Towards Partial Fulfillment for the Award of
Bachelor of Engineering in Computer Science Engineering

Guided by:

Mr. Gajendra Rajput

Assistant Professor,

CSE , AITR, Indore

Submitted by:

Kanishk Chouhan(0827CS201112)

Khushboo Sen(0827CS201114)

Mahak Soni(0827CS201127)

Mayank Solanki(0827CS201134)

Acropolis Institute of Technology & Research, Indore

Jan - June 2023

EXAMINER APPROVAL

The Project entitled "***Gesture Recognition using OpenCV***" submitted by **Kanishk Chouhan (0827CS201112), Khushboo Sen (0827CS201114), and Mahak Soni (0827CS201127), Mayank Solanki (0827CS201134),** have been examined and are hereby approved for partial fulfillment for the award of a ***Bachelor of Technology degree in Computer Science Engineering*** discipline, for which it has been submitted. It is understood that by this approval the undersign does not necessarily endorse or approve any statement made, opinion expressed, or conclusion drawn therein, but approves the project only for the purpose for which it has been submitted.

(Internal Examiner)
Date:

GUIDE RECOMMENDATION

This is to certify that the work embodied in this project entitled "***Gesture Recognition using OpenCV***" submitted by ***Kanishk Chouhan (0827CS201112), Khushboo Sen (0827CS201114), Mahak Soni (0827CS201127), Mayank Solanki (0827CS201134)*** is a satisfactory account of the bonafide work done under the supervision of Mr. Gajendra Rajput is recommended towards partial fulfillment for the award of the Bachelor of Technology (Computer Science Engineering) degree by Rajiv Gandhi Proudyogiki Vishwavidhyalaya, Bhopal.

(Project Guide)

(Project Coordinator)

(Dean Academics)

STUDENTS UNDERTAKING

This is to certify that project entitled "***Gesture Recognition using OpenCV***" has been developed by us under the supervision **of Mr. Gajendra Rajput**. The whole responsibility of work done in this project is ours. The sole intention of this work is only for practical learning and research.

We further declare that to the best of our knowledge, this report does not contain any part of any work which has been submitted for the award of any degree either in this University or in any other University / Deemed University without proper citation and if the same work found then we are liable for explanation to this.

Kanishk Chouhan (0827CS201112)
Khushboo Sen (0827CS201114)
Mahak Soni (0827CS201127)
Mayank Solanki (0827CS201134)

Acknowledgment

We thank the almighty Lord for giving me the strength and courage to sail out through the tough and reach on shore safely.

We would be failing in our duty if do not acknowledge the support and guidance received from **Dr S C Sharma**, Director, AITR, Indore whenever needed. We take the opportunity to convey my regards to the management of Acropolis Institute, Indore for extending academic and administrative support and providing me with all the necessary facilities for project to achieve our objectives.

We owe a debt of sincere gratitude, deep sense of reverence and respect to our guide and mentor **Dr. Kamal Kumar Sethi,** Hod, AITR, Indore for his motivation, sagacious guidance, constant encouragement, vigilant supervision and valuable critical appreciation throughout this project work, which helped us to successfully complete the project on time.

We express profound gratitude and heartfelt thanks to **Mr. Gajendra Rajput**, Ass. Professor & Head CSE, AITR Indore for his support, suggestion, and inspiration for carrying out this project.We are very much thankful to other faculty and staff members of IT Dept, AITR Indore for providing me with all support, help, and advice during the project.

We are grateful to **our parents** and **family members** who have always loved and supported us unconditionally. To all of them, we want to say "Thank you", for being the best family that one could ever have and without whom none of this would have been possible.

Kanishk Chouhan (0827CS201112)

Khushboo Sen (0827CS201114)

Mahak Soni (0827CS201127)

Mayank Solanki (0827CS201134)

List of Figures

Table of Contents

Chapter 1. Introduction

Introduction

For any user, authenticated by face recognition, few gestures or voice commands could be defined for frequently used tasks- save, exit, print, screen-lock, screen unlock, system shut down, system restart. Save, print and exit operations are context sensitive meaning that it is applicable for current application. For example if word document is open and the gesture for save is done then the document will saved, if print voice command is given then printer dialog will open etc. Similarly a gesture or a voice command could be defined for close/exit which will close the current application. If no application is opened then it will work as system shut down. It is similar to Alt+F4 key press functionality on windows PC.

1.1 Overview

A gesture recognition system is a technology that uses computer vision and machine learning algorithms to interpret and classify human hand gestures in real-time. The system typically consists of a camera or sensor that captures the hand movements, an image processing module that extracts features from the captured images, and a machine learning model that classifies the gestures based on these features.

The basic workflow of a gesture recognition system involves several steps:

Data collection: The system needs to collect a large dataset of hand gestures that will be used to train the machine learning model.

Data pre-processing: The collected data needs to be processed to remove noise and artifacts that can affect the accuracy of the system. This can involve techniques like image filtering, segmentation, and normalization.

Feature extraction: The pre-processed data is then analyzed to extract features that can be used to classify the hand gestures. This can involve techniques like edge detection, contour analysis, and motion analysis.

Gesture classification: The extracted features are fed into a machine learning model that has been trained on the dataset of hand gestures. The model classifies the hand gestures into different categories based on the learned patterns and features.

Gesture recognition: Once the hand gestures have been classified, the system can take appropriate actions based on the recognized gesture. This can involve controlling a virtual avatar, navigating through a user interface, or triggering specific actions in an application.

Gesture recognition systems have a wide range of applications, including gaming, healthcare, education, and more. They can be used to create more intuitive and natural interfaces between humans and machines, and to improve accessibility for individuals with disabilities.

1.2 Background and Motivation

Hand gesture recognition system is a technology that enables machines to interpret and respond to human hand gestures. The primary motivation behind the development of such systems is to create a more natural and intuitive interface between humans and machines.

With the rise of virtual and augmented reality technologies, there is an increasing need for hand gesture recognition systems that can accurately track and interpret hand movements in real-time. This can be used for a wide range of applications, including gaming, education, healthcare, and more.

Additionally, hand gesture recognition systems have the potential to improve accessibility for individuals with disabilities, such as those who are unable to use traditional input devices like a mouse or keyboard.

The development of hand gesture recognition systems involves complex image processing and machine learning algorithms that enable the system to accurately recognize and classify different hand gestures. This requires a significant amount of data collection, annotation, and training, as well as ongoing optimization and refinement of the system.

Overall, the motivation behind the development of hand gesture recognition systems is to create a more natural and intuitive interface between humans and machines, improve accessibility for individuals with disabilities, and enable new and innovative applications in various fields.

1.3 Problem Statement and Objectives

For any user, authenticated by face recognition, few gestures or voice commands could be defined for frequently used tasks- save, exit, print, screen-lock, screen unlock, system shut down, system restart. Save, print and exit operations are context sensitive meaning that it is applicable for current application. For example if word document is open and the gesture for save is done then the document will saved, if print voice command is given then printer dialog will open etc. Similarly a gesture or a voice command could be defined for close/exit which will close the current application. If no application is opened then it will work as system shut down. It is similar to Alt+F4 key press functionality on windows PC.

The goal of this project is to develop a software system that allows the user/people to perform some frequently performed functions using gestures and voice commands and thus saving a lot of time. Gesture recognition can be seen as a way for computers to begin to understand human body language. Compared to the primitive user interfaces, such as keyboard and mouse, it builds a richer bridge between the computers and humans.

1.4 Scope of the Project

The scope for hand gesture recognition is quite broad, as it has applications in various fields. Here are some of the key areas where hand gesture recognition systems can be used:

Human-computer interaction: Gesture recognition systems can be used to create more intuitive and natural interfaces between humans and computers, allowing users to interact with devices through hand gestures instead of traditional input devices like a keyboard or mouse.

Gaming: Gesture recognition can be used to create more immersive gaming experiences, allowing players to control the game characters and objects through hand gestures.

Healthcare: Hand gesture recognition can be used to develop rehabilitation tools for patients with motor disabilities, allowing them to control medical devices and perform exercises with the help of hand gestures.

Education: Gesture recognition can be used to create interactive learning experiences, allowing students to control educational software and interact with digital content through hand gestures.

Robotics: Gesture recognition can be used in robotics applications, allowing robots to recognize and respond to human hand gestures, improving their ability to interact with humans.

Security: Gesture recognition can be used as a biometric authentication method, allowing users to authenticate themselves through hand gestures, providing a more secure and convenient authentication method.

Overall, the scope for hand gesture recognition is quite broad, and the technology has the potential to transform the way humans interact with machines in various fields.

1.5 Report Structure

The project **Hand Gesture Recognition system** is primarily concerned with the interaction of human with computer without any physical intervention and whole project report is categorized into five chapters.

Chapter 1: Introduction- introduces the background of the problem followed by rationale for the project undertaken. The chapter describes the objectives, scope and applications of the project. Further, the chapter gives the details of team members and their contribution in development of project which is then subsequently ended with report outline.

Chapter 2: Review of Literature- explores the work done in the area of Project undertaken and discusses the limitations of existing system and highlights the issues and challenges of project area. The chapter finally ends up with the requirement identification for present project work based on findings drawn from reviewed literature and end user interactions.

Chapter 3: Proposed System - starts with the project proposal based on requirement identified, followed by benefits of the project. The chapter also illustrate software engineering paradigm used along with different design representation. The chapter also includes block diagram and details of major modules of the project. Chapter also gives insights of different type of feasibility study carried out for the project undertaken. Later it gives details of the different deployment requirements for the developed project.

Chapter 4: Implementation - includes the details of different Technology/ Techniques/ Tools/ Programming Languages used in developing the Project. The chapter also includes the different user interface designed in project along with their functionality. Further it discuss the experiment results along with testing of the project. The chapter ends with evaluation of project on different parameters like accuracy and efficiency.

Chapter 5: Conclusion - Concludes with objective wise analysis of results and limitation of present work which is then followed by suggestions and recommendations for further improvement.

Chapter 2 . Review of Literature

2.1 Preliminary Investigation

2.1.1 Current System

- **Microsoft Kinect:** Kinect is a gesture recognition system developed by Microsoft for use with their Xbox gaming console. It uses an RGB camera and an infrared sensor to track the user's body movements and hand gestures, allowing them to control games and other applications without the need for a traditional input device.

- **Leap Motion:** Leap Motion is a gesture recognition system that uses two small infrared cameras to track hand and finger movements with high precision. It can be used for a wide range of applications, including gaming, virtual reality, and education.

- **Google Soli:** Soli is a radar-based gesture recognition system developed by Google. It uses a small chip that emits radar signals and detects the reflected signals to track hand gestures. It is primarily used in mobile devices and wearables, allowing users to control devices through hand gestures.

2.2 Limitations of Current System

- **Microsoft Kinect:** The Kinect system requires a significant amount of space and a clear line of sight to the user, making it less suitable for small or cluttered environments. It also has some limitations in terms of tracking finer hand gestures and finger movements.

- **Leap Motion:** The Leap Motion system requires a relatively high level of processing power and can be affected by environmental factors such as ambient lighting and reflections. It may also have difficulty tracking hand gestures in certain positions or orientations.

- **Google Soli**: The Soli system has limited range and accuracy compared to some other gesture recognition systems. It is also still a relatively new technology and may have limited support from third-party developers.

2.3 Requirement Identification and Analysis for Project

Requirement analysis is an important step in the development of any system, including a hand gesture recognition system. Here are some of the key requirements that should be considered during the analysis phase:

Accuracy: The hand gesture recognition system should be able to accurately recognize and distinguish between different hand gestures, even in challenging conditions such as low lighting or background clutter.

Speed: The system should be able to recognize hand gestures quickly and respond in real-time to user input.

Robustness: The system should be robust enough to handle variations in hand size, shape, and orientation, as well as variations in lighting and environmental conditions.

Ease of use: The system should be easy for users to understand and use, with clear and intuitive gestures that can be easily learned and performed.

Compatibility: The system should be compatible with a wide range of devices and platforms, including mobile devices, computers, and virtual reality systems.

Privacy and security: The system should be designed with privacy and security in mind, with appropriate safeguards in place to protect user data and prevent unauthorized access.

Flexibility: The system should be flexible enough to support a wide range of applications, from gaming and entertainment to healthcare and education.

Cost-effectiveness: The system should be cost-effective, with a balance between the cost of hardware, software, and development time.

These are just a few of the key requirements that should be considered during the requirement analysis phase of developing a hand gesture recognition system. By carefully considering these requirements and other factors, developers can create a system that is accurate, efficient, and well-suited to the needs of its users.

2.3.1 Conclusion

Hand gesture recognition is of great importance for human computer interaction (HCI) because of its extensive applications in virtual reality and sign language recognition etc. Human hand is very smaller with very complex articulations comparing with the entire human body and therefore errors can be easily affected. There can be no doubt that voice assistants are, and will continue to become, a great feat of human ingenuity and they are already creeping into our lives in some shape or form. With the eventual roll-out of 5G and the improvement in machine learning voice assistants may be setting themselves up to be a tool we cannot live without. However, before we get to that stage, there are hurdles to cross which include heavy investment, improvement in the technology and confidence from consumers that this device that is in their lives does not pose a risk to their privacy.

Chapter 3 . Proposed System

Proposed System

3.1 The Proposal

Over a period, the generation has witnessed a paradigm shift- a switch that altered the entire practice that we employ to manage and operate electronic devices. The digital era has welcomed gesture recognition software to meet the dynamic need of technological innovation, which can recognize hand and face movements and perform the assigned/asked task after that. The software interprets human gestures using programmed mathematical algorithms. It has facilitated every arena lying in the digital sphere of the world, such as a graphic designer who can correctly make changes in an image without using the keyboard or mouse. An engineer can zoom in or zoom out on a 3-Dimensional representation of any model only by the hand gestures when the device is in their hand. Gesture recognition technology has a lot of benefits by which the human can direct their businesses and personal life.

Gesture recognition consists of three basic levels:

Detection: With the help of a camera, a device detects hand or body movements, and a machine learning algorithm segments the image to find hand edges and positions.

Tracking: A device monitors movements frame by frame to capture every movement and provide accurate input for data analysis.

Recognition: The system tries to find patterns based on the gathered data. When the system finds a match and interprets a gesture, it performs the action associated with this gesture. Feature extraction and classification in the scheme below implements the

recognition functionality.

3.2 Benefits of the Proposed System

The current system had a lot of challenges that are overcome by this system :

- **Saves time:** By using this system lots of time can be saved**.**
- **Easy to use:** It is very easy to use.

3.3 Block Diagram

Figure 3.1

3.4 Feasibility Study

A feasibility study is an analysis of how successfully a system can be implemented, accounting for factors that affect it such as economic, technical and operational factors to determine its potential positive and negative outcomes before investing a considerable amount of time and money into it .

3.4.1 Technical

Hand gesture recognition is one of the most viable and popular solution for improving human computer interaction. In the recent years it has become very popular due to its use in gaming devices like Xbox, PS4, and other devices like laptops, smart phones, etc. Hand Detection & gesture recognition has usage in various applications like medicine, accessibility support etc. In this paper, we would like to propose on how to develop a hand gesture recognition simulation using OpenCV and python 2.7.

Prerequisites for this project:

1. Python – 3.x (we used Python 3.8.8 in this project)
2. OpenCV – 4.5 Run “*pip install opencv-python*” to install OpenCV.
3. cvzone-1.5.2 Run “*pip install cvzone*” to install cvzone

3.5 Design Representation

3.5.1 UML diagrams

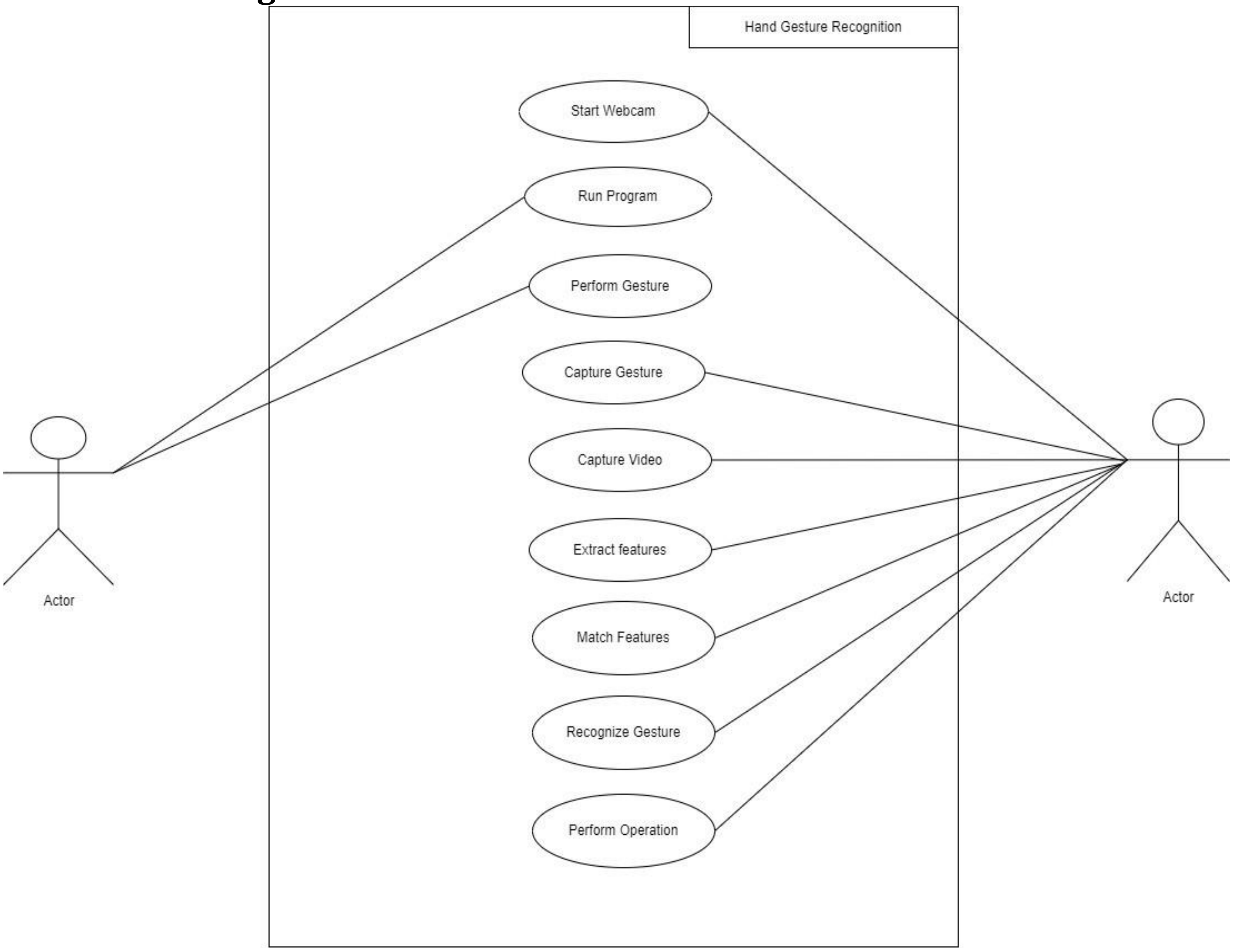

Figure 3.2 use case diagram

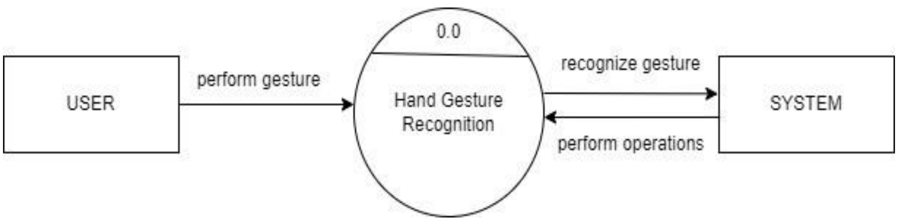

Figure 3.3. level 0 DFD

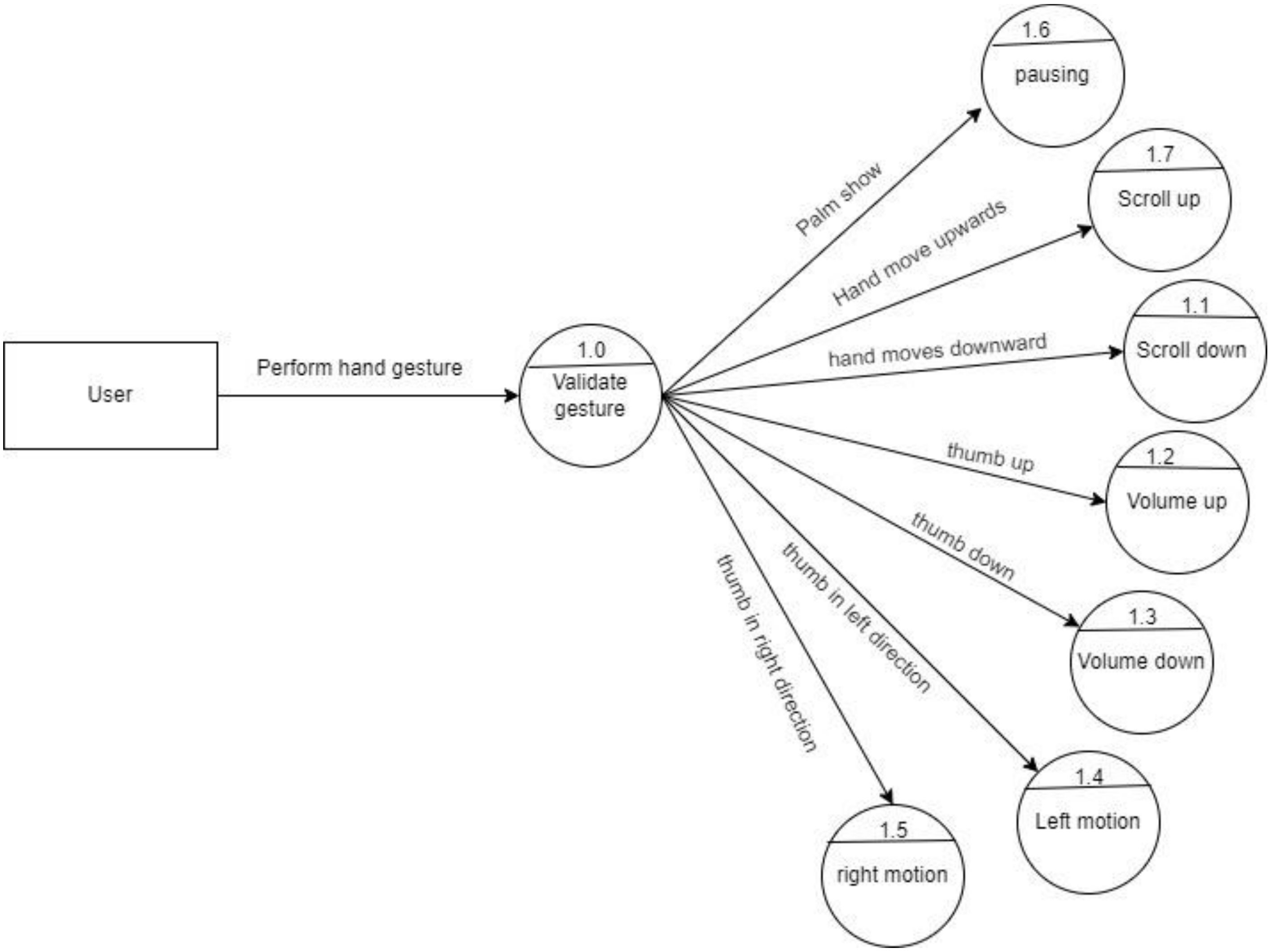

Figure 3.4. level 1 DFD

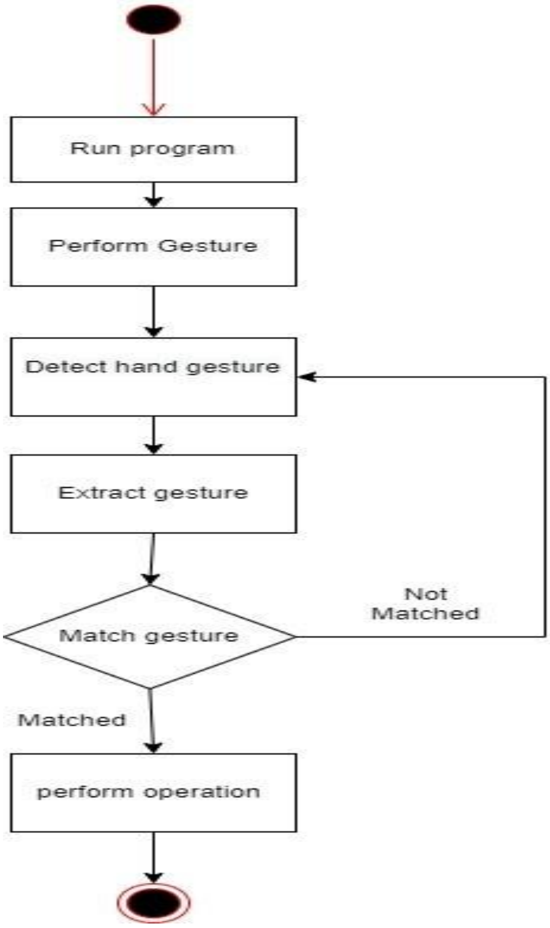

Figure 3.5. state diagram

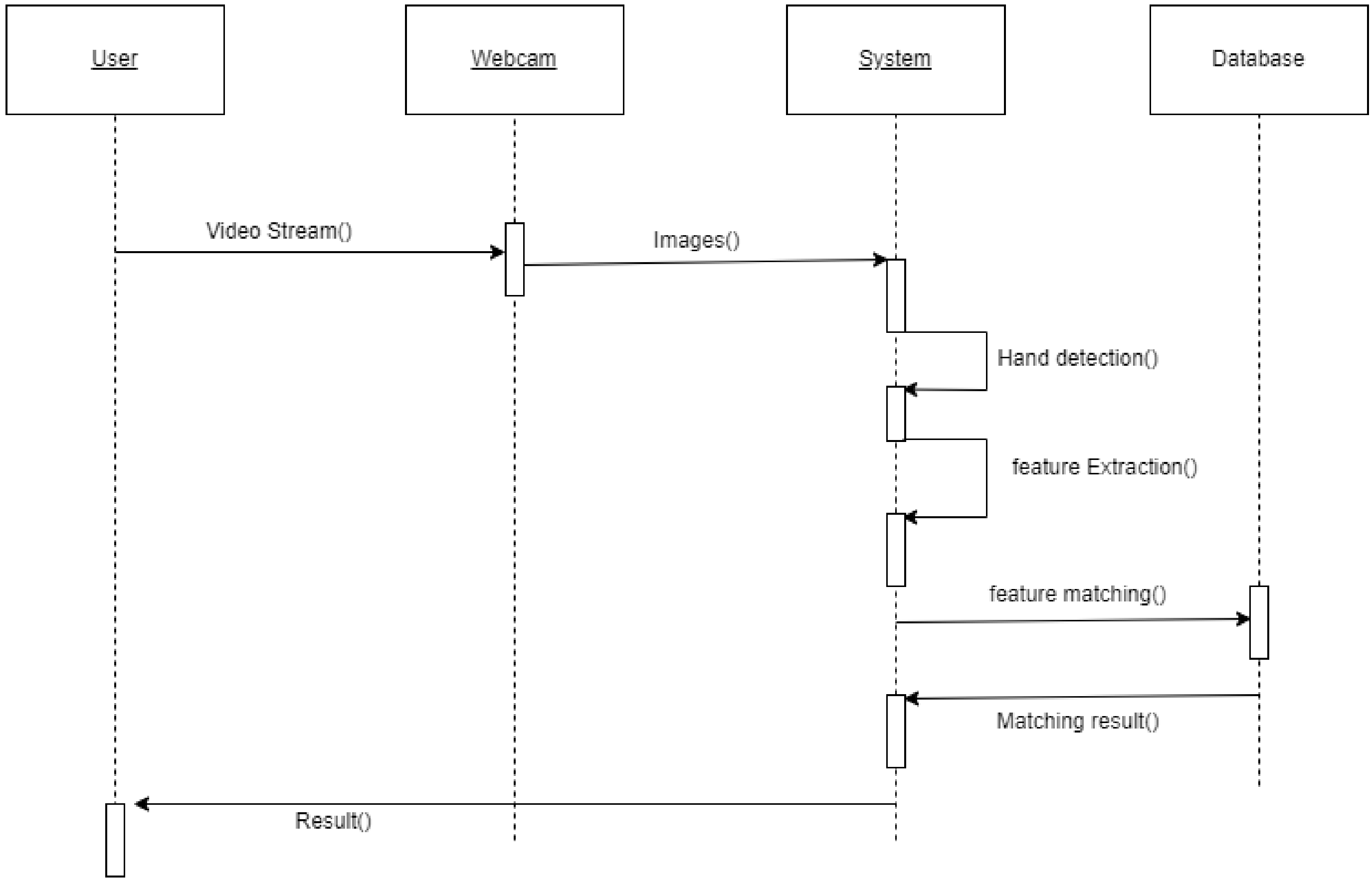

Figure 3.6. Sequence diagram

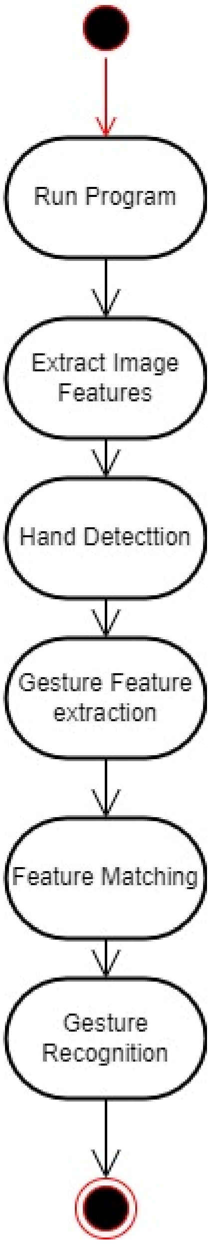

Figure 3.7. Activity diagram

3.6 Deployment Requirements

There are various requirements (hardware, software and services) to successfully deploy the system. These are mentioned below :

3.6.1 Hardware

- System
- 4GB of RAM
- 500GB of Hard disk

3.6.2 Software

- Python Idle
- Library Packages

Chapter 4 . Implementation

Implementation

Hand gesture recognition technique steps vary from simple to complex applications. Generally, the steps are usually divided as the following: first hand gesture frame acquisition, then hand tracking, then feature extraction, and finally classification to reach the output gesture.

4.1 Technique Used

4.1.1 Gesture acquisition methods

Image hand gesture acquisition, as illustrated in from Microsoft(2019)is to capture the human hand gesture image by the computer (Ananya, Anjan Kumar & Kandarpa Kumar, 2015). This could be done using vision-based recognition where no special gadgets are required and a web camera or a depth camera is used, furthermore special tools can be utilized such as wired or wireless gloves that detect the movements of the user hand, and motion sensing input devices (Kinect from Microsoft, Leap Motion, etc.) that capture the hand gestures and motions.

4.2 Tools Used

Gesture recognition features:

- Higher accuracy
- Higher stability
- Quicker time to unlock a device

The major application areas of gesture recognition in the current scenario are:

- Automotive sector
- Consumer electronics sector
- Transit sector
- Gaming sector
- To unlock smartphones
- Defense
- Home automation
- Automated sign language translation

4.2.1 Draw.io

Designed by Seibert Media, draw.io is proprietary software for making diagrams and charts. The software lets you choose from an automatic layout function, or create a custom layout. They have a large selection of shapes and hundreds of visual elements to make your diagram or chart one-of-a-kind. The drag-and-drop feature makes it simple to create a great looking diagram or chart.

Draw.io has options for storing saved charts in the cloud, on a server, or network storage at a data center, depending on your needs

4.3 Language Used

The program is developed by using python programming language with the help of additional libraries such as OpenCV.

Python

Python is a computer programming language often used to build websites and software, automate tasks, and conduct data analysis. Python is a general-purpose language, meaning it can be used to create a variety of different programs and isn't specialized for any specific problems. This versatility, along with its beginner-friendliness, has made it one of the most-used programming languages today.

Stack Overflow's 2022 Developer Survey revealed that Python is the fourth most popular programming language, with respondents saying that they use Python almost 50 percent of the time in their development work. Survey results also showed that Python is tied with Rust as the most-wanted technology, with 18% percent of developers who aren't using it already saying that they are interested in learning Python .

OpenCV

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being an Apache 2 licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

4.5.1 Strategy Used

- Gesture recognition can be conducted with techniques from computer vision and image processing.
- The literature includes ongoing work in the computer vision field on capturing gestures or more general human pose and movements by cameras connected to a computer.

Chapter 5.Conclusion
Conclusion

5.1 Conclusion

Hand gesture recognition is of great importance for human computer interaction (HCI) because of its extensive applications in virtual reality and sign language recognition etc. Human hand is very smaller with very complex articulations comparing with the entire human body and therefore errors can be easily affected. There can be no doubt that voice assistants are, and will continue to become, a great feat of human ingenuity and they are already creeping into our lives in some shape or form. With the eventual roll-out of 5G and the improvement in machine learning voice assistants may be setting themselves up to be a tool we cannot live without. However, before we get to that stage, there are hurdles to cross which include heavy investment, improvement in the technology and confidence from consumers that this device that is in their lives does not pose a risk to their privacy.

5.2 Limitations of the Work

Unfortunately, like other image processing problems, hand tracking and segmentation in a cluttered background is a critical problem in hand gesture recognition. In most occasions, the background is not simple. Also, the background may contain other body parts captured by the camera.

5.3 Suggestion and Recommendations for Future Work

Suggestions for future work is that, various hand gestures can be recognized and applied as input to the computer. The hand gestures

representing numbers can also be converted into commands to perform related tasks in real time. Enhancing the recognition capability for various lightning conditions, which is encountered as a challenge in this project can be worked upon in future.

Bibliography

[1] https://www.geeksforgeeks.org/?newui
[2] https://www.javatpoint.com/
[3] https://developers.google.com/mediapipe/solutions/vision/gesture_recognizer/python/
[4] https://pypi.org/project/opencv-python/
[5] https://www.ibm.com/topics/convolutional-neural-networks/

Guide Interaction Sheet

Project Work/Phases	Marks %	Submission week& Tentative Dates*
Synopsis/ Proposal	10	July 8, 2022
Synopsis Presentation	10	July 15, 2022
Design Phase (ERD, DFD, Use-case & other UML diagrams)	10	August 16, 2022
Implementation Demo + Report	20	September 12, 2022
Project based paper	10	September 22,2022
Video and poster	10	September 22,2022
Final Project Submission (Presentation + Demo + Report)	20	October 03, 2022
Attendance in project lab	10	Attendance =>70% fetch max
Total Marks	**100**	

Source code

```
from cvzone.HandTrackingModule import HandDetector
import cv2
import os
import numpy as np

# Parameters
width, height = 1280, 720
gestureThreshold = 300
folderPath = "presentation"

# Camera Setup
cap = cv2.VideoCapture(0)
cap.set(3, width)
cap.set(4, height)

# Hand Detector
detectorHand = HandDetector(detectionCon=0.8, maxHands=1)

# Variables
imgList = []
delay = 30
buttonPressed = False
counter = 0
drawMode = False
imgNumber = 0
delayCounter = 0
annotations = [[]]
annotationNumber = -1
annotationStart = False
hs, ws = int(120 * 3), int(213 * 3)  # width and height of small image

# Get list of presentation images
pathImages = sorted(os.listdir(folderPath), key=len)
print(pathImages)
#add ppt to png converter module
while True:
    # Get image frame
    success, img = cap.read()
    img = cv2.flip(img, 1)
    pathFullImage = os.path.join(folderPath, pathImages[imgNumber])
    imgCurrent = cv2.imread(pathFullImage)

    # Find the hand and its landmarks
    hands, img = detectorHand.findHands(img)  # with draw
    # Draw Gesture Threshold line
    cv2.line(img, (0, gestureThreshold), (width, gestureThreshold), (0, 255, 0), 10)

    if hands and buttonPressed is False:  # If hand is detected

        hand = hands[0]
        cx, cy = hand["center"]
        lmList = hand["lmList"]  # List of 21 Landmark points
        fingers = detectorHand.fingersUp(hand)  # List of which fingers are up

        # Constrain values for easier drawing
        xVal = int(np.interp(lmList[8][0], [width // 2, width], [0, width]))
        yVal = int(np.interp(lmList[8][1], [150, height - 150], [0, height]))
        indexFinger = xVal, yVal

        if cy <= gestureThreshold:  # If hand is at the height of the face
            if fingers == [1, 0, 0, 0, 0]:
                print("Left")
```

```
                buttonPressed = True
                if imgNumber > 0:
                    imgNumber -= 1
                    annotations = [[]]
                    annotationNumber = -1
                    annotationStart = False
            if fingers == [0, 0, 0, 0, 1]:
                print("Right")
                buttonPressed = True
                if imgNumber < len(pathImages) - 1:
                    imgNumber += 1
                    annotations = [[]]
                    annotationNumber = -1
                    annotationStart = False

        if fingers == [0, 1, 1, 0, 0]:
            cv2.circle(imgCurrent, indexFinger, 12, (0, 0, 255), cv2.FILLED)

        if fingers == [0, 1, 0, 0, 0]:
            if annotationStart is False:
                annotationStart = True
                annotationNumber += 1
                annotations.append([])
            print(annotationNumber)
            annotations[annotationNumber].append(indexFinger)
            cv2.circle(imgCurrent, indexFinger, 12, (0, 0, 255), cv2.FILLED)

        else:
            annotationStart = False

        if fingers == [0, 1, 1, 1, 0]:
            if annotations:
                annotations.pop(-1)
                annotationNumber -= 1
                buttonPressed = True

    else:
        annotationStart = False

    if buttonPressed:
        counter += 1
        if counter > delay:
            counter = 0
            buttonPressed = False

    for i, annotation in enumerate(annotations):
        for j in range(len(annotation)):
            if j != 0:
                cv2.line(imgCurrent, annotation[j - 1], annotation[j], (0, 0, 200),
12)

    imgSmall = cv2.resize(img, (ws, hs))
    h, w, _ = imgCurrent.shape
    imgCurrent[0:hs, w - ws: w] = imgSmall

    cv2.imshow("Slides", imgCurrent)
    cv2.imshow("Image", img)

    # waitkey frame waiting time
    key = cv2.waitKey(1)
    if key == ord('q'):
        break
```