# A

**MINI PROJECT REPORT**

## ON

## VOLUME AND BRIGHTNESS CONTROL USING HAND GESTURES

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

## BACHELOR OF TECHNOLOGY

IN

## COMPUTER SCIENCE AND ENGINEERING

## Submitted by

Under the Guidance

Of

**Department of CSE**



## Department of Computer Science and Engineering TEEGALA KRISHNA REDDY ENGINEERING COLLEGE

**(An Autonomous Institution)**

**Medbowli, Meerpet, Saroornagar, Hyderabad – 500097**

**(Affiliated to JNTUH, Approved by AICTE, Accredited by NBA & NAAC ‘A’) (2019-2023)**

  **TEEGALA KRISHNA REDDY ENGINEERING COLLEGE**

**(Sponsored by TKR Educational Society)**

**Approved by AICTE, Affiliated by JNTUH, Accredited by NBA & NAAC-A)**

Medbowli, Meerpet, Saroornagar, Hyderabad – 500 097.

Phone: 040-24092838 Fax: +91-040-24092555

E-mail: [tkrec@rediffmail.com](mailto:tkrec@rediffmail.com) Website: [www.tkrec.ac.in](http://www.tkrec.ac.in)

**Department of Computer Science & Engineering**

# College code: R9

# CERTIFICATE

This is to certify that the Mini Project report on **“Volume and Brightness Control by using Hand Gestures ”** is a bonafide work carried out by

in partial fulfillment for the requirement of the award of B.Tech degree in Computer Science and Engineering, Teegala Krishna Reddy Engineering College, Hyderabad, affiliated to Jawaharlal Nehru Technological University, Hyderabad under my guidance and supervision.

The result of investigation enclosed in this report have been verified and found satisfactory. The results embodied in the project work have not been submitted to any other University for the award of any degree.

## INTERNAL GUIDE HEAD OF DEPARTMENT

D.Sravani Dr.CH.V.Phani Krishna

Assistant Professor Professor

……………………………….. ……………………………..

## EXTERNAL EXAMINER PRINCIPAL

Dr. K. Venkata MuraliMohan

Professor

………………..……… ………………………

  **TEEGALA KRISHNA REDDY ENGINEERING COLLEGE**

**(Sponsored by TKR Educational Society)**

**Approved by AICTE, Affiliated by JNTUH, Accredited by NBA & NAAC-A)**

Medbowli, Meerpet, Saroornagar, Hyderabad – 500 097.

Phone: 040-24092838 Fax: +91-040-24092555

E-mail: [tkrec@rediffmail.com](mailto:tkrec@rediffmail.com) Website: [www.tkrec.ac.in](http://www.tkrec.ac.in)

**Department of Computer Science & Engineering**

# College code: R9

# DECLARATION

We hereby declare that the Mini Project report entitled **“VOLUME AND BRIGHTNESS CONTROL USING HAND GESTURES”** is done under the guidance of **D. Sravani (Assistant Professor)**. **,** Department of Computer Science and Engineering, Teegala Krishna Reddy Engineering College, is submitted in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering** from **Jawaharlal Nehru Technological University**, Hyderabad.

This is a record of bonafide work carried out by us in **Teegala Krishna Reddy Engineering College** and the results embodied in this project have not been reproduced or copied from any source.

**Submitted by**

# 

# ACKNOWLEDGEMENT

The satisfaction and euphoria that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose encouragement and guidance have crowned our efforts with success.

We extend my deep sense of gratitude to **Dr.K.Venkata Murali Mohan**, **Principal** Teegala Krishna Reddy Engineering College, Meerpet, for permitting me to undertake this project.

We are indebted to **Dr.CH.V.Phani Krishna, Professor & Head of the Department**, Computer Science and Engineering, Teegala Krishna Reddy Engineering College, Meerpet for his support and guidance throughout our project.

We are indebted to our guide **D. Sravani (Assistant Professor)**, Computer Science Engineering, Teegala Krishna Reddy Engineering College, Meerpet for his support and guidance throughout our project.

We are indebted to the project coordinators **A. DIVYA SREE,** Assistant professor, and **Y. SHIVA SREE**, Assistant Professor, Computer Science and Engineering, Teegala Krishna Reddy Engineering College, Meerpet for his support and guidance throughout our project.

Finally, we express thanks to one and all that have helped me in successfully completing this technical seminar. Further I would like to thank my family and friends for their moral support and encouragement.

**By**

**ABSTRACT**

We are developing a volume and brightness controller in which we are using hand gestures as the input to control the system , Opencv module is basically used in this implementation to control the gesture.

Hand gesture recognition system has developed excessively in the recent years, reason being its ability to cooperate with machine successfully. Gestures are considered as the most natural way for communication among human and PCs in virtual framework. We often use hand gestures to convey something as it is non-verbal communication which is free of expression. In our system, we used background subtraction to extract hand region. In this application, our PC's camera records a live video, from which a preview is taken with the assistance of its functionalities or activities.

This system basically uses the web camera to record or capture the images /videos and accordingly on the basis of the input, the volume and brightness of the system is controlled by this application. The main function is to increase and decrease the volume and brightness of the system. The project is implemented using Python, OpenCV.

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO** | **NAME OF THE FIGURE** | **PAGE.NO** |
| 4.1 | System Architecture | 15 |
| 4.1.2 | Flow of the Project | 16 |
| 4.2.1 | Use Case Diagram | 17 |
| 4.2.2 | Class Diagram | 18 |
| 4.2.3 | Sequence Diagram | 19 |
| 4.2.6 | Activity Diagram | 20 |
| 5.1 | Python Installation | 21 |
| 5.1.1 | Pycharm download | 22 |
| 5.1.2 | Pycharm | 23 |
| 5.2.1 | Convolution operation | 26 |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **TABLE NO** | **NAME OF THE TABLE** | **PAGE NO.** |
| 2.1 | LITERATURE SURVEY | 13 |

|  |  |
| --- | --- |
| **TABLE CONTENTS** |  |
| TITLE | Page.no |
| **CHAPTER 1: INTRODUCTION** | 9-12 |
| 1.1 PROBLEM STATEMENT | 9 |
| * 1. OBJECTIVES | 10 |
| * 1. MOTIVATIONS | 10 |
| * 1. EXISTING SYSTEM | 11 |
| 1.5 PROPOSED SYSTEM | 11 |
| 1.6 SCOPE | 12 |
| **CHAPTER 2: LITERATURE SURVEY** | 13 |
| CHAPTER **3: SYSTEM ANALYSIS** | 14 |
| **CHAPTER 4: SYSTEM DESIGN** | 15-20 |
| 4.1. SYSTEM ARCHITECTURE | 15 |
| 4.1.1. ARCHTECTURE DESCRIPTIONS | 15 |
| 4.1.2. FLOW OF THE PROJECT | 16 |
| 4.2. UML DAIGRAMS | 16-20 |
| 4.2.1. USE CASE DAIGRAM | 17 |
| 4.2.2. CLASS DAIGRAM | 18 |
| 4.2.3. SEQUENCE DAIGRAM | 19 |
| 4.2.4. ACTIVITY DAIGRAM | 20 |
| **CHAPTER 5: IMPLEMENTION** | 21-36 |
| 5.1. ENVIRONMENTAL SETUP | 21 |
| 5.2. ALGORITHM | 26 |
| 5.3. MODULE DESCRIPITION | 28 |
| **CHAPTER 6: SYSTEM TEST** | 37 |
| 6.1. TESTING | 37 |
| 6.1.1. UNIT TESTING | 37 |
| 6.1.2. INTEGRATION TESTING | 37 |
| 6.1.3. ACCEPCTANCE TESTING | 37 |
| **CHAPTER 7: CODING** | 38-39 |
| **CHAPTER 8: OUTPUT SCREENS** | 40-42 |
| **CHAPTER 9: CONCLUSION** | 43 |
| **CHAPTER 10: FUTURE ENHANCEMENT** | 44 |
| **CHAPTER 11: REFERENCE** | 45 |

# CHAPTER 1

# INTRODUCTION

Hand gestures are unprompted and also robust transmission mode for Human Computer Interaction (HCI). Keyboard, mouse, joystick or touch screen are some input devices for connection with the computer but they don't provide appropriate interface whereas, the current system will contain either desktop or laptop interface in which hand gesture can be done by wearing data gloves or web camera used for snapping hand image. The first step towards this gesture recognition is hand capturing and analyzing. Sensors are used in Data-Glove methods for initializing fingers movement and other sensor will program hand movements. In comparison the vision-based method only needs a camera and hence identifying the actual interaction between human and computer without using any other devices. The challenges of this system are constant background, sometimes person and lighting also. Different procedure and algorithms which are used in this system are elaborated here along with the recognition techniques. The method of searching a connecting region in the picture with particular specification, being it color or intensity, where a pattern and algorithm is adjustable is known as segmentation. Vision Based method requires a web camera, so that one can realize natural interaction between humans and computer without using any other devices. The challenging part in these systems is background images or videos which is recorded or captured during taking the inputs i.e. hand gesture by the user, also sometime lightning effect the quality of the input taken which creates the problem in recognizing the gestures. Process to find a connected region within the image with some of the property such as color ,intensity and a relationship between pixels i.e. pattern is termed as segmentation. And have used some important packages which have OpenCv-python, tensorflow, numpy, mediapipe, imutils, scipy, numpy

* 1. **PROBLEM STATEMENT**

We are developing a model of human computer interaction(HCI) which will able to control our laptop or desktop volume and brightness by using hand gestures. Gesture recognition has been reshaped for different research applications being it face movements gestures or whole body gestures (Dong, Yan, & Xie, 1998). Few applications has developed and created a hard requirement for this kind of recognition system (Dong et al., 1998). Coming to static recognition system, it is a design recognition problem, for instance, an important part of design recognition preprocessing level, called, feature extraction, must be controlled or managed before any standard pattern or design recognition process can be applied on it.

* 1. **OBJECTIVES**

The objective of this project:

* To recognize hands and fingers movements to control volume and brightness.
* To identify index and thumb finger.
* To calculate the distance between the index and thumb and performs the task.
* To identify left and right hand (left: volume , right: brightness)
* To develop an interface which will capture human hand gesture dynamically and will control the volume level.
  1. **MOTIVATION**

Recently, strong efforts have been carried out to develop intelligent and natural interfaces between users and computer-based systems based on human gestures. Gestures provide an intuitive interface to both humans and computers. Thus, such gesture-based interfaces can not only substitute the common interface devices but can also be exploited to extend their functionality. A robot is usually an electro-mechanical machine that can perform tasks automatically. Some robots require some degree of guidance, which may be done using a remote control or with a computer interface. Robots have evolved so much and are capable of mimicking humans that they seem to have a mind of their own. An important aspect of a successful robotic system is Human-Machine interaction. In the early years, the only way to communicate with a robot was to program which required extensive hard work. With the development in science and robotics, Gesture Recognition came into life

Hand gestures is the powerful communication medium for Human Computer Interaction (HCI).Several input devices are available for interaction with computer, such as keyboard, mouse, joystick and touch screen, but these devices does not provide easier way to communicate. Hand gestures will provide an easier and friendly way to communication with computer.

* 1. **EXISTING SYSTEM**
* A DSP, DM6437 of Texas Instruments, is used for our portable hand gesture recognition system. For the real-time hand gesture sensing and recognition system, we propose a finger skin pixel algorithm to quickly and easily distinguish the hand in a complex image and use the region of interest to reduce the amount of computation. The hand trace direction can be found easily using a hand gesture center point in our system. Finally, our system is applied to TV channel and volume control using hand gestures and hand tracing.
* Applied scaled normalization for gesture recognition based on brightness factor matching. The input image with is segmented using thresholding technique where the background is black. Two methods are used for extraction the features; firstly, by using the edge mages, and secondly by using normalized features where only the brightness values of pixels are calculated and other black pixels are neglected to reduce the length of the feature vector . The database consists of 6 different gestures, 10 samples per gesture are used, 5 samples for training and 5 samples for testing.

**DRAWBACKS OF EXISTING SYSTEM**

* Time consuming
* Expensive
* It is having high Computational consuming
* It uses the complete frame or complete web cam
* Using lowest pixels cameras
  1. **PROPOSED SYSTEM**

We are developing a volume and brightness controller in which we are using hand gestures as the input to system.

A vision-based hand Gesture system that does not require any special markers or gloves and can operate in real-time on a commodity PC with low-cost cameras.

* 1. **SCOPE**
* The scope of this project is unlimited and can be used in organizations, buildings like schools, offices. These gestures are chosen because they are commonly used to communicate and can thus be used in various applications, such as, a virtual mouse that can perform six tasks (Open, Close, Cut, Paste, Maximize, Minimize) for a given application.
* Hand gesture analysis can be divided into two main approaches, namely, glove based analysis, vision-based analysis . The glove-based approach employs sensors (mechanical or optical) attached to a glove that acts as transducer of finger flexion into electrical signals to determine hand posture. The relative position of the hand is determined by an additional sensor. This sensor is normally a magnetic or an acoustic sensor attached to the glove

**CHAPTER 2**

**LITERATURE SURVEY**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S. No** | **AUTHORNAME** | **PROBLEMS**  **IDENTIFIED** | **TECHNIQUES**  **USED** | **ACCURACY** | **DRAWBACKS** |
| 1. | Mahmoud E and Bernd M | To Recognize the Isolated and Meaningful Hand Gesture | Hidden Markov  Model | 93.84 | It is having high Computational consuming |
| 2. | Hasan | Gesture Recognition based on brightness factor matching | Trimming and Scaling Normalization Technique | 95% | It uses the complete frame or complete web cam. |
| 3 | Robust | Gesture recognition for robotic control | Dynamic Gesture recognition | 93.2% | Using lowest pixels cameras |

**CHAPTER 3**

**SYSTEM ANALYSIS**

1. **HARDWARE REQUIREMENTS**

* Processor : I3/Intel Processor
* Hard Disk : 160GB
* RAM : 8Gb

1. **SOFTWARE REQUIREMENTS**

* Operating System : Windows 7/8/10
* IDE : Pycharm
* Libraries Used : Mediapipe, OpenCV
* Technology : Python 3.6+
* Accessories : Webcam.

**CHAPTER 4**

**SYSTEM DESIGN**

**4.1 SYSTEM ARCHITECTURE**

IMAGES CAPTURED

**44**

HAND DETECTION

PRE- PROCEESING OF IMAGES

PERFORMING REQUIRED TASK

OUTPUT

FINGER MOVEMENT

Fig -4.1 System Architecture

**4.1.1 ARCHITECTURE OF DISCRIPTION**

* In the first step the user's image will be captured through the camera.
* The captured image will be pre-processed.
* The features of the pre-processed image will be extracted.
* Then, the training phase will be done.
* Now ,an identification process takes place i.e., if the trained image is matched with the image present in the database then access will be granted.
* If the trained image does not match with the image present in the database then a buzzer will be alarmed and an alert sms will be sent to the admin.

**4.1.2 FLOW OF THE PROJECT**

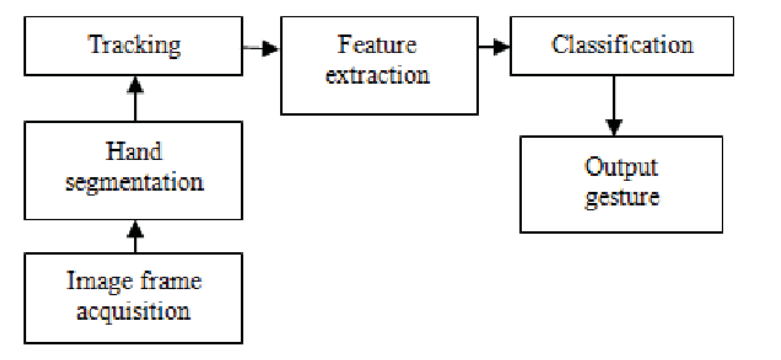
****

Fig-4.1.2 FLOW OF THE PROJECT

**4.2 UML DIAGRAMS**

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: A Metamodel and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects

* + 1. **USE CASE DIAGRAM**
* A use case diagram in the Unified Modeling Language (UML) is a type of behavioral 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.

|  |
| --- |
|  |

Fig-4.2.1 USE CASE DIAGRAM

* + 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 which information.

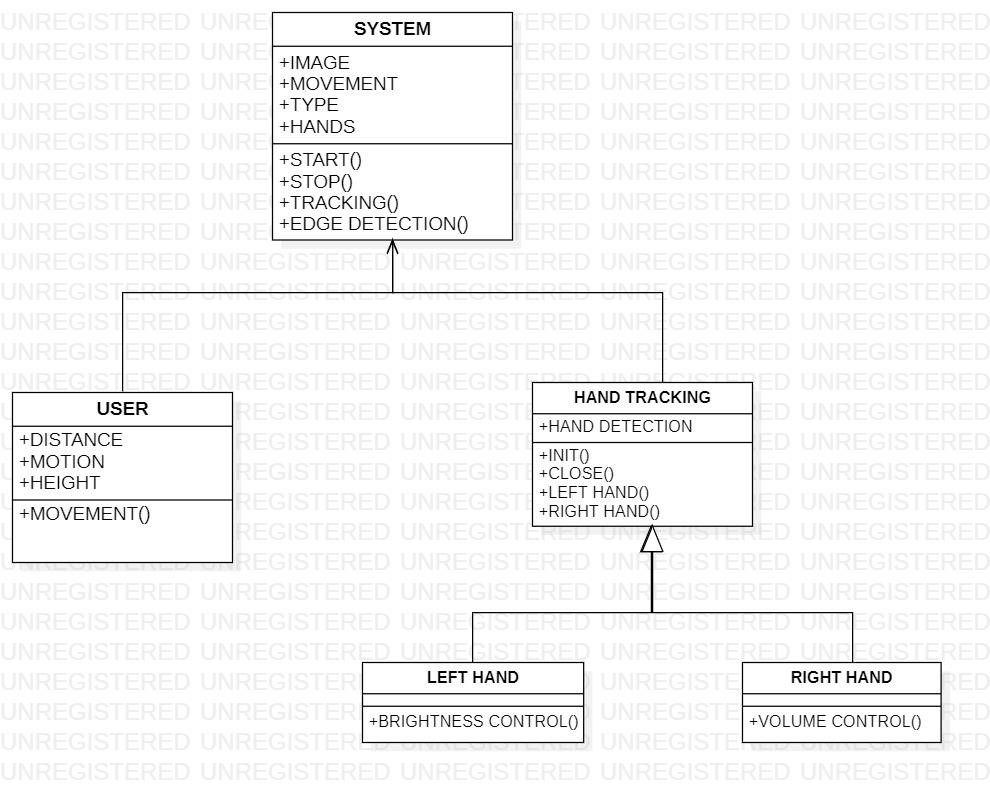
****

Fig-4.2.2 CLASS DIAGRAM

* + 1. **SEQUENCE DIAGRAM**
* A sequence diagram in Unified Modeling 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.

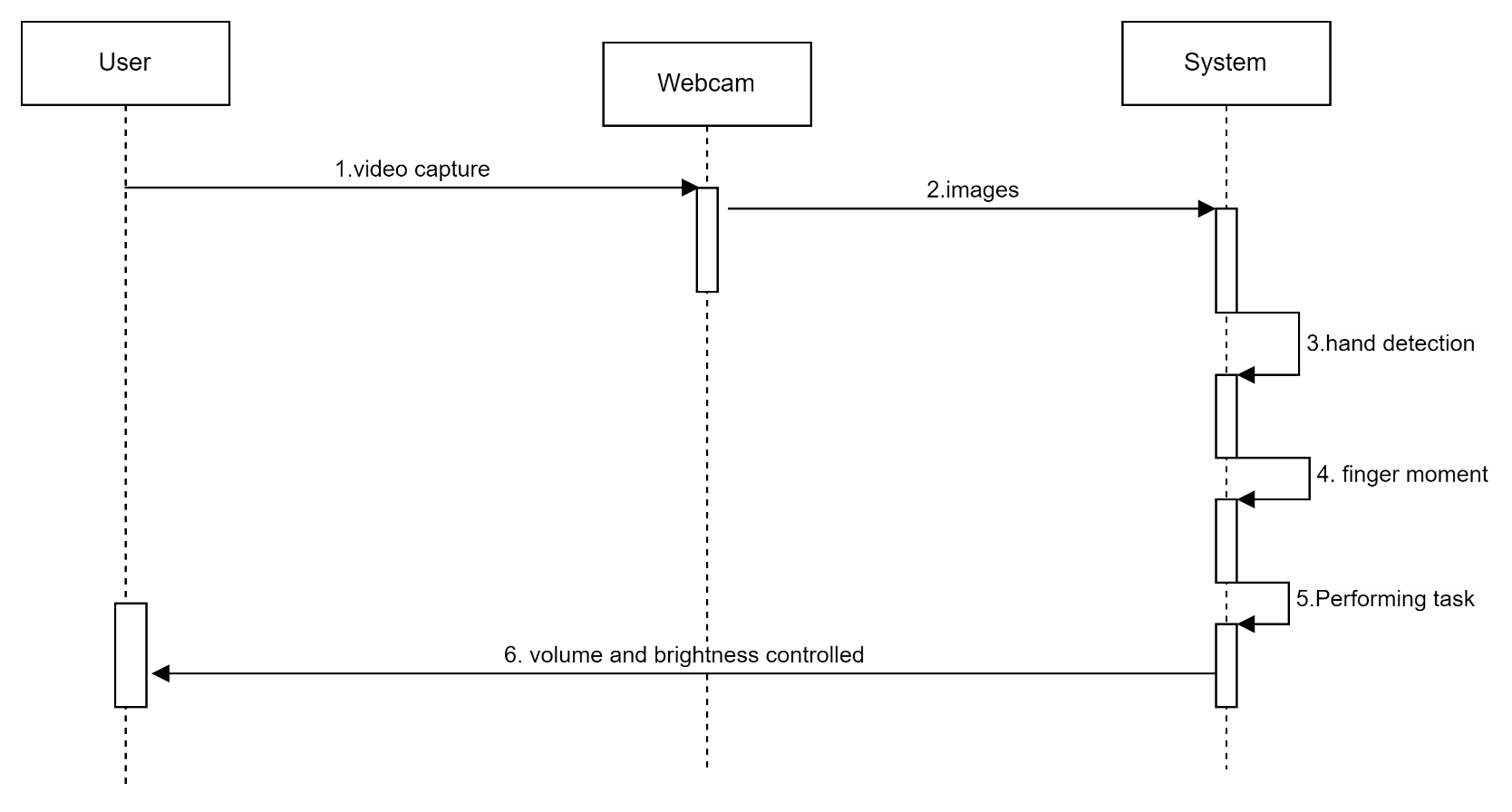
****

Fig-4.2.3 SEQUENCE DIAGRAM

**4.2.6 ACTIVITY DIAGRAM**

* Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

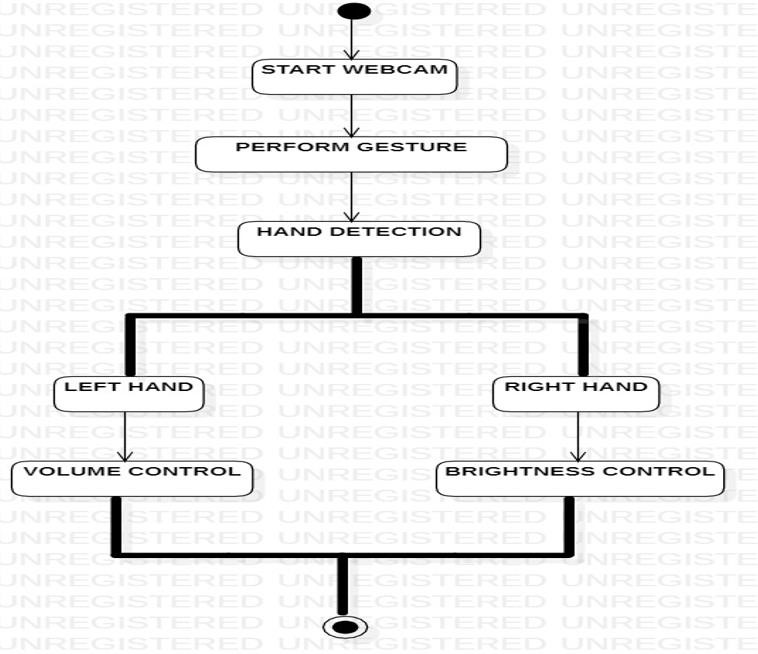
****

Fig-4.2.6 ACTIVITY DIAGARM

**CHAPTER 5**

**IMPLEMENTATION**

**5.1 ENVIRONMENTAL SETUP**

Installing Python:

1. To download and install Python visit the official website of Python https://www.python.org/downloads/ and choose your version.

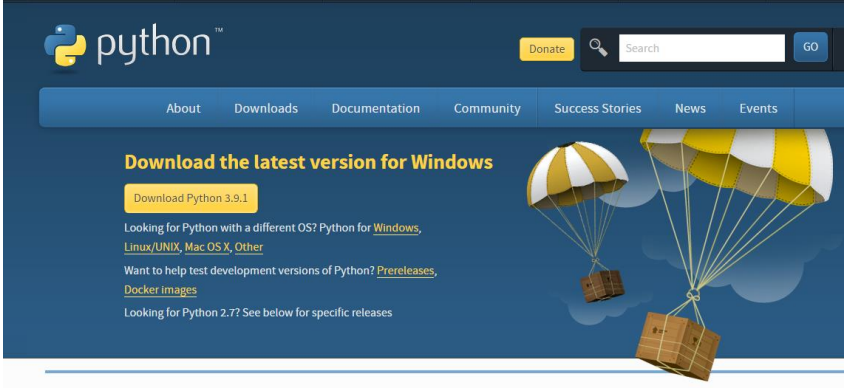
****

FIG-5.1 PYTHON INSTALLATION

1. Once the download is complete, run the exe for install Python. Now click on Install Now.
2. You can see Python installing at this point.
3. When it finishes, you can see a screen that says the Setup was successful. Now click on "Close".

Installing PyCharm:

1. To download PyCharm visit the

website https://www.jetbrains.com/pycharm/download/ and Click the

"DOWNLOAD" link under the Community Section.

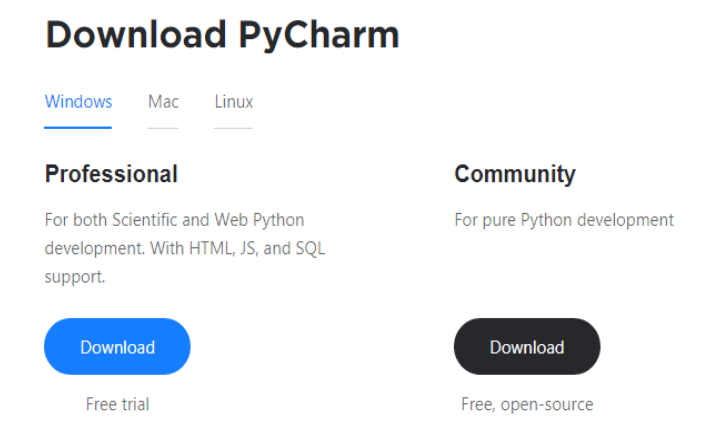
****

FIG-5.1.1 PYCHARM DOWNLOAD

1. Once the download is complete, run the exe for install PyCharm. The setup wizard should have started. Click “Next”.
2. On the next screen, Change the installation path if required. Click “Next”.
3. On the next screen, you can create a desktop shortcut if you want and click on “Next”.
4. Choose the start menu folder. Keep selected JetBrains and click on “Install”.
5. Wait for the installation to finish.
6. Once installation finished, you should receive a message screen that PyCharm is installed. If you want to go ahead and run it, click the “Run PyCharm Community Edition” box first and click “Finish”.
7. After you click on "Finish," the Following screen will appear .

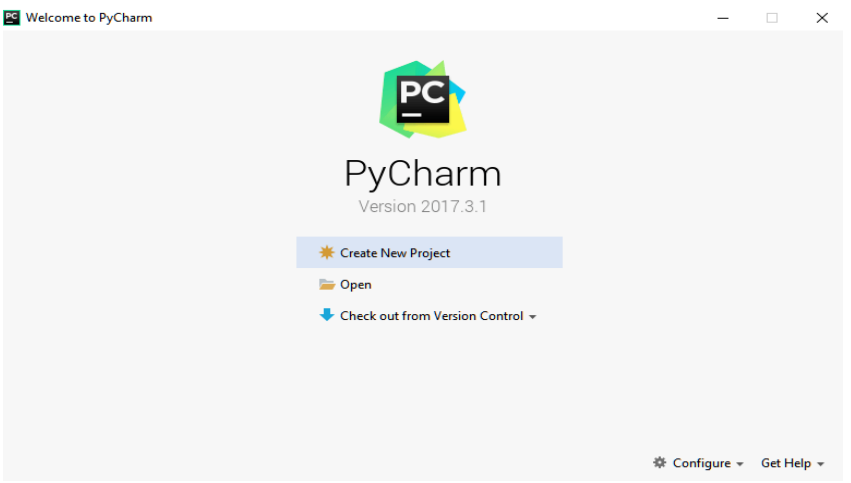
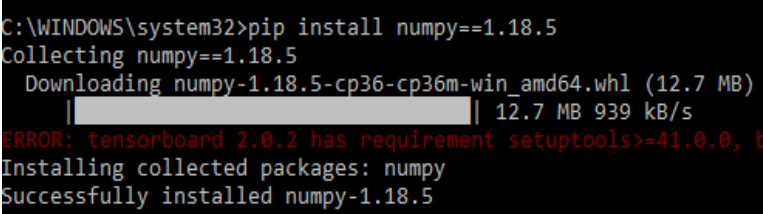


FIG-5.1.2 PYCHARM

1. You need to install some packages to execute your project in a proper way.
2. Open the command prompt/ anaconda prompt or terminal as administrator.
3. The prompt will get open, with specified path, type “pip install package name” which you want to install (like numpy, pandas, seaborn, scikit-learn, matplotlib. pyplot)

Ex: pip install numpy 

1. Install pycaw:

Latest stable release:

pip install pycaw

Development branch:

pip install <https://github.com/AndreMiras/pycaw/archive/develop.zip>

System requirements:

choco install visualcpp-build-tools

1. Install mediapipe:
2. Download the latest protoc win64 zip from the Protobuf GitHub repo, unzip the file,

and copy the protoc.exe executable to a preferred location . Please ensure that is added into the Path environment variable.

1. Activate a Python virtual environment.

**$ python3 -m venv mp\_env && source mp\_env/bin/activate**

1. In the virtual environment, go to the MediaPipe repo directory.
2. Install the required Python packages.

**(mp\_env)mediapipe$ pip3 install -r requirements.txt**

1. Generate and install MediaPipe package.

**mp\_env)mediapipe$ python3 setup.py gen\_protos**

**(mp\_env)mediapipe$ python3 setup.py install --link-opencv**

1. **SYSTEM**
   1. **Capture Images:** The camera in our device is used for this project. It detects our

hand with points in it so as it can see the distance between our thumb finger tip and index finger tip. The distance between the points 4 and 8 is directly proportional to the volume of device. Capture the images using open CV platform

**1.2 Image detection:**

* Detect hand landmarks
* Calculate the distance between thumb tip and index finger tip.
* Map the distance of thumb tip and index finger tip with volume range. For my case, distance between thumb tip and index finger tip was within the range of 30 – 350 and the volume range was from -63.5 – 0.0.
* In order to exit press ‘Spacebar'
* After the hand detection in captured frames, the next step is to control the system volume depending on direction of hand movement. The hand movement direction is determined by generating and locating the bounding box on the detected hand

1. **USER**

* Detect hand landmarks
* Calculate the distance between thumb tip and index finger tip.
* Map the distance of thumb tip and index finger tip with volume range. For my case, distance between thumb tip and index finger tip was within the range of 30 – 350 and the volume range was from -63.5 – 0.0.
* In order to exit press ‘Spacebar'
* **2.1 Enter First Name:** The user needs to enter his/her first name.
* **2.2 Enter Last Name**: The user need to enter his/her last name
* **2.3 Capture Images:**

The user requests the application to capture images of his/her face without disguise.

* **2.4 View Results:**

Whenever unknown person is identified, our system will buzzer an alarm with loud

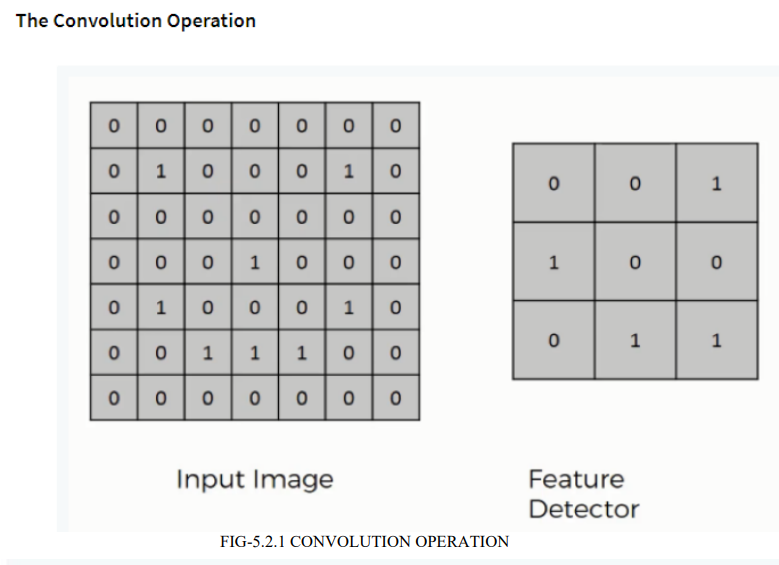
voice and message also sent to the application owner as an alert

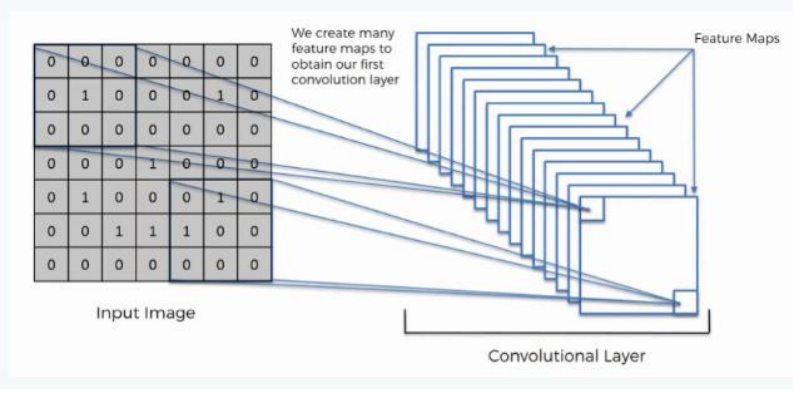
**5.2 ALGORITHM**

**1.Conventional Neural Network**

**Step1: Convolutional operation**

The first building block in our plan of attack is convolution operation. In this step, we will touch on feature detectors, which basically serve as the neural network's filters. We will also discuss feature maps, learning the parameters of such maps, how patterns are detected, the layers of detection, and how the findings are mapped out



****

**Step (1b): Relu Layer**

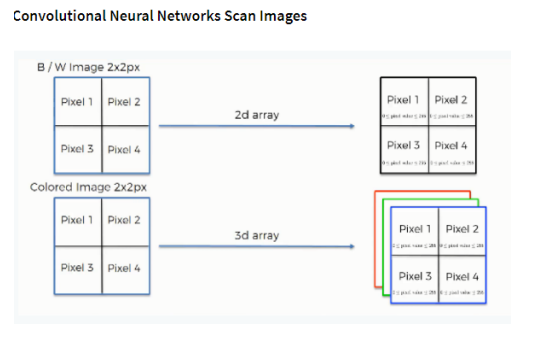
The second part of this step will involve the Rectified Linear Unit or Relook. We will

cover Relook layers and explore how linearity functions in the context of Convolutional

Neural Networks.

Not necessary for understanding CNN's, but there's no harm in a quick lesson to improve

your skills.

****

**Step 2: Pooling Layer**

In this part, we'll cover pooling and will get to understand exactly how it generally works.

Our nexus here, however, will be a specific type of pooling; max pooling. We'll cover various

approaches, though, including mean (or sum) pooling. This part will end with a demonstration

made using a visual interactive tool that will definitely sort the whole concept out for you.

**Step 3: Flattening**

This will be a brief breakdown of the flattening process and how we move from pooled to

flattened layers when working with Convolutional Neural Networks.

**Step 4: Full Connection**

In this part, everything that we covered throughout the section will be merged together. By

learning this, you'll get to envision a fuller picture of how Convolutional Neural Networks

operate and how the "neurons" that are finally produced learn the classification of images.

**Summary**

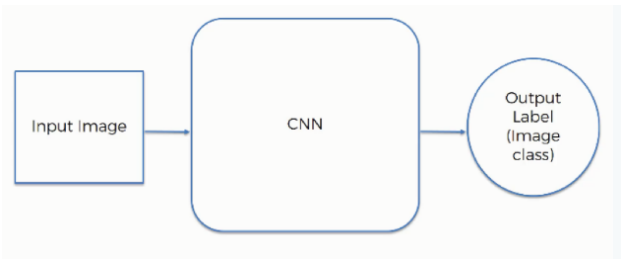
In the end, we'll wrap everything up and give a quick recap of the concept covered in the

section. If you feel like it will do you any benefit (and it probably will), you should check

out the extra tutorial in which Soft ax and Cross-Entropy are covered. It's not mandatory

for the course, but you will likely come across these concepts when working with

Convolutional Neural Networks and it will do you a lot of good to be familiar with them.



**5.3 MODULE DISCRIPTION**

* OpenCV
* Numpy
* Mediapipe

**OpenCV:**

OpenCV is a library of programming capacities mostly focused on ongoing PC vision.

Initially created by Intel, it was later bolstered by Willow Garage then Itseez . The library

is cross-stage and free for use under the open-source BSD permit.

OpenCV underpins a few models from profound learning structures like TensorFlow,

Torch, PyTorch (in the wake of changing over to an ONNX model) and Caffe as indicated

by a characterized rundown of upheld layers. It advances Open Vision Capsules. , which is

a versatile configuration, perfect with every other organization. Authoritatively propelled

in 1999 the OpenCV venture was at first an Intel Research activity to propel CPU-

concentrated applications, some portion of a progression of undertakings including

constant beam following and 3D show dividers. The principle supporters of the undertaking

remembered various improvement specialists for Intel Russia, just as Intel's Performance

Library Team.

OpenCV is written in C++ and its essential interface is in C++, yet it despite everything

holds a less far reaching however broad more seasoned C interface. There are ties in

Python, Java and MATLAB/OCTAVE. Since variant 3.4, OpenCV.js is a JavaScript

official for chose subset of OpenCV capacities for the web stage.

The entirety of the new turns of events and calculations in OpenCV are presently evolved

in the C++ interface.

OpenCV runs on the accompanying work area working frameworks: Windows, Linux,

macOS, FreeBSD, NetBSD, OpenBSD. OpenCV runs on the accompanying portable

working frameworks: Android, iOS, Maemo, BlackBerry 10. The client can get official

discharges from SourceForge or take the most recent sources from GitHub. OpenCV

utilizes CMake.

**Advantages:**

* OpenCV is accessible liberated from cost.
* As OpenCV library is written in C/C++ it is very quick.
* Low RAM use .
* It is versatile as OpenCV can run on any gadget which runs on C

As a human, handling pictures is a characteristic procedure as we cooperate with

numerous individuals and it is simple for us to perceive in our day by day lives. PCs, then

again, need to follow by their own interesting procedures so as to break down huge measures

of media information. All together for profound learning PC vision to flourish, thousands on a

great many photographs, recordings, and different pictures should be incorporated for a viable

AI to get valuable

The PC arranges a picture dependent on pixel esteems, it isolates a picture into an enormous

framework of boxes - also called pixels - and allots a number to each case .

* 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 a BSD-licensed product,

OpenCV makes it easy for businesses to utilize and modify the code.

* The library has more than 2500 optimized algorithms, which includes a comprehensive

set of both classic and state-of-the-art computer vision and machine learning algorithms. learning algorithms. identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc. OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 18 million. The library is used extensively in companies, research groups and by governmental bodies

* Along with well-established companies like Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota that employ the library, there are many startups such as Applied Minds, VideoSurf, and Zeitera, that make extensive use of OpenCV. OpenCV’s deployed uses span the range from stitching streetview images together, detecting intrusions in surveillance video in Israel, monitoring mine equipment in China, helping robots navigate and pick up objects at Willow Garage, detection of swimming pool drowning accidents in Europe, running interactive art in Spain and New York, checking runways for debris in Turkey, inspecting labels on products in factories around the world on to rapid face detection in Japan.
* It has C++, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. OpenCV leans mostly towards real-time vision applications and takes advantage of MMX and SSE instructions when available. A full-featured CUDA and OpenCL interfaces are being actively developed right now. There are over 500 algorithms and about 10 times as many functions that compose or support those algorithms. OpenCV is written natively in C++ and has a templated interface that works seamlessly with STL containers.

**NumPy:**

* NumPy is a Python library, including support for enormous, multi-dimensional clusters and frameworks, alongside a huge assortment of significant level numerical capacities to work on these exhibits. The predecessor of NumPy, Numeric, was initially made by Jim Hugunin with commitments from a few different designers. In 2005, Travis Oliphant made NumPy by consolidating highlights of the contending Numarray into Numeric, with broad alterations. NumPy is open-source programming and has numerous givers. Such clusters can likewise be sees into memory cradles distributed by C/C++, Cython, and Fortran expansions to the CPython mediator without the need to duplicate information around, giving a level of similarity with existing numerical libraries. This usefulness is misused by the SciPy bundle, which wraps various such libraries . NumPy has worked in help for memory-mapped ndarrays.
* NumPy focuses on the CPython reference execution of Python, which is a non-streamlining bytecode translator. Numerical calculations composed for this rendition of Python regularly run much more slow than assembled counterparts. NumPy addresses the gradualness issue incompletely by giving multidimensional clusters and capacities and administrators that work productively on exhibits, requiring changing some code, generally internal circles utilizing NumPy.
* Utilizing NumPy in Python gives usefulness practically identical to MATLAB since they are both deciphered, and the two of them permit the client to compose quick projects as long as most activities take a shot at exhibits or frameworks rather than scalars. In examination, MATLAB brags a huge number extra tool compartments, remarkably Simulink, while NumPy is inherently coordinated with Python, an increasingly present day and complete programming language. Additionally, reciprocal Python bundles are accessible; SciPy is a library that includes more MATLAB-like usefulness and Matplotlib is a plotting bundle that gives MATLAB-like plotting usefulness. Inside, both MATLAB and NumPy Inside, depend on BLAS and LAPACK for effective direct variable based math calculations.
* Python ties of the broadly utilized PC vision library OpenCV use NumPy exhibits to store and work on information. Since pictures with various channels are essentially spoken to as three-dimensional clusters, ordering, cutting or concealing with different exhibits are exceptionally proficient approaches to get to explicit pixels of a picture. The NumPy cluster as widespread information structure in OpenCV for pictures, extricated include focuses, channel pieces and a lot more limitlessly streamlines the programming work process and troubleshooting.
* The center usefulness of NumPy is its "ndarray", for n-dimensional exhibit, information structure. Rather than Python's worked in list information structure, these clusters are homogeneously composed: all components of a solitary exhibit must be of a similar kind.
* Such clusters can likewise be sees into memory cradles distributed by C/C++, Cython, and Fortran expansions to the CPython mediator without the need to duplicate information around, giving a level of similarity with existing numerical libraries. This usefulness is misused by the SciPy bundle, which wraps various such libraries . NumPy has worked in help for memory-mapped ndarrays.

****

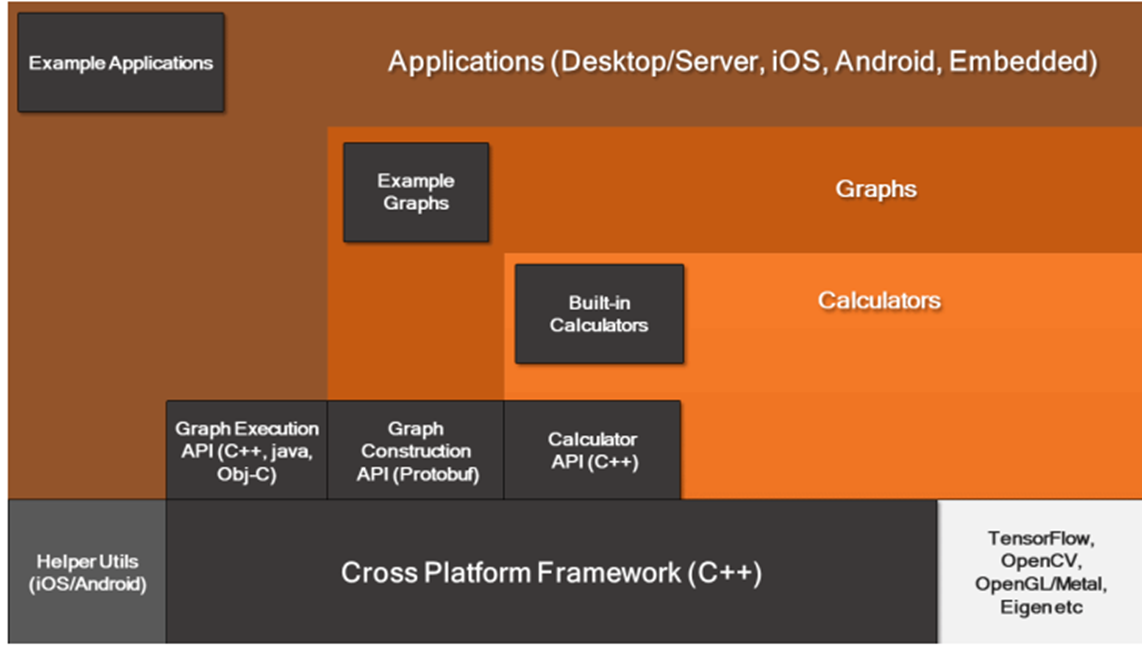
**Mediapipe**

* MediaPipe is a Framework for building machine learning pipelines for processing time-series data like video, audio, etc. This cross-platform Framework works in Desktop/Server, Android, iOS, and embedded devices like Raspberry Pi and Jetson Nano.
* MediaPipe Toolkit comprises the **Framework** and the **Solutions.** The following diagram shows the components of the MediaPipe Toolkit.
* **Framework**: The Framework is written in C++, Java, and Obj-C, which consists of the following APIs.

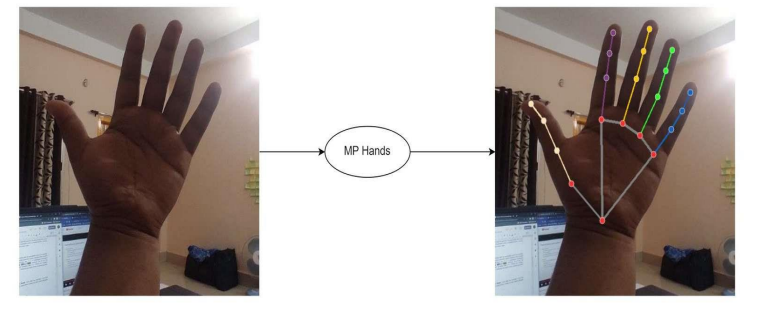
1. **Calculator API** (C++).

2**. Graph construction** API (Protobuf).

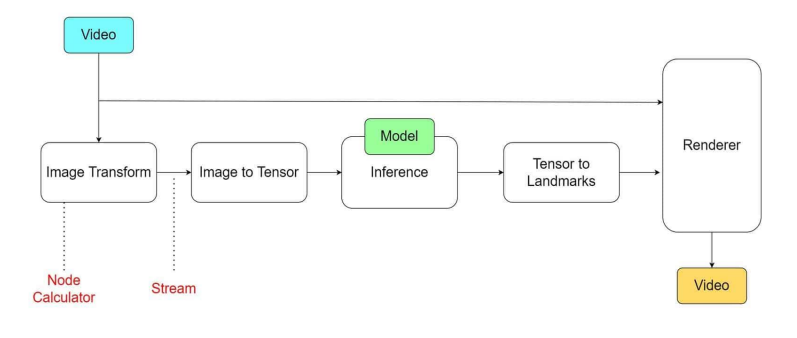
3**. Graph Execution API** (C++, Java, Obj-C)



* **Graphs:** The MediaPipe perception pipeline is called a Graph. Let us take the example of the first solution, Hands. We feed a stream of images as input which comes out with hand landmarks rendered on the images.



• The flow chart below represents the MP (Abbr. MediaPipe) hand solution graph.



MediaPipe hands solution graph

* In computer science jargon, a graph consists of Nodes connected by Edges. Inside the MediaPipe Graph, the nodes are called **Calculators**, and edges are called **Streams.** Every stream carries a sequence of **Packets** that have ascending time stamps.
* In the image above, we have represented Calculators with rectangular blocks and Streams using arrows.
* **MediaPipe Solutions :** Solutions are open-source pre-built examples based on a specific pre-trained TensorFlow or TFLite model. You can check Solution specific models here. MediaPipe Solutions are built on top of the Framework. Currently, it provides sixteen solutions as listed below.

1. Face Detection

2. Face Mesh

3. Iris

4. Hands

5. Pose

6. Holistic

7. Selfie Segmentation

8. Hair Segmentation

9. Object Detection

10. Box Tracking

11. Instant Motion Tracking

12. Objectron

13. KNIFT

14. AutoFlip

15. MediaSequence

16. YouTube 8M

• The solutions are available in C++, Python, JavaScript, Android, iOS, and Coral. **As of now, majority of the solutions are available only in C++ (except KNIFT and IMT) followed by Android, with Python not too far behind.**

• The other wrapper languages, too, are growing fast with a very active development state. As you can see, even though MediaPipe Framework is cross-platform, that does not imply the same for the solutions. MediaPipe is currently at alpha version 0.7. We can expect the solutions to get more support with the beta releases. Following are some of the solutions provided by MediaPipe.

**• Synchronization and Performance Optimization**

MediaPipe supports multimodal graphs. To speed up the processing, different calculators run in separate threads. For performance optimization, many built-in calculators come with options for GPU acceleration. Working with time series data must be in proper synchronization; otherwise, the system will break. The graph ensures this so that flow is handled correctly according to the timestamps of packets. The Framework handles synchronization, context sharing, and inter-operations with CPU calculators

**CHAPTER 6**

**SYSTEM TESTING**

**6.1 TESTING**

**6.1.1 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 objective**

* All field entries must work properly.
* Pages must be activated from the identified link.
* Pages must be activated from the identified link.

**Features to be tested**

* Verify that the entries are of the correct format
* All links should take the user to the correct page.

**6.1.2 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.

**6.1.3 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 defect encounters.

**CHAPTER 7**

**CODING**

import cv2  
import numpy as np  
import math  
from ctypes import cast,POINTER  
from comtypes import CLSCTX\_ALL  
from pycaw.pycaw import AudioUtilities,IAudioEndpointVolume  
import screen\_brightness\_control as sbc  
import threading  
  
from handLmModel import handDetector  
  
vidObj = cv2.VideoCapture(0)  
vidObj.set(cv2.CAP\_PROP\_FRAME\_WIDTH,1280)  
vidObj.set(cv2.CAP\_PROP\_FRAME\_HEIGHT,720)  
  
handlmsObj = handDetector(detectionCon=0.7)  
  
devices = AudioUtilities.GetSpeakers()  
interface = devices.Activate(IAudioEndpointVolume.\_iid\_,CLSCTX\_ALL,None)  
volume = cast(interface,POINTER(IAudioEndpointVolume))  
volRange = volume.GetVolumeRange()  
minVolume = volRange[0]  
maxVolume = volRange[1]  
  
minBrightness = 0  
maxBrightness = 100  
  
def setVolume(dist):  
 vol = np.interp(int(dist), [35, 215], [minVolume, maxVolume])  
 volume.SetMasterVolumeLevel(vol, None)  
  
def setBrightness(dist):  
 brightness = np.interp(int(dist), [35, 230], [minBrightness, maxBrightness])  
 sbc.set\_brightness(int(brightness))  
  
while True:  
 \_,frame = vidObj.read()  
 frame = cv2.flip(frame,1)  
 frame = handlmsObj.findHands(frame)  
 lndmrks = handlmsObj.findPosition(frame,draw=False)  
 if lndmrks:  
 # print(lndmrks[4],lndmrks[8])  
  
 xr1,yr1 = lndmrks[1][4][1],lndmrks[1][4][2]  
 xr2,yr2 = lndmrks[1][8][1],lndmrks[1][8][2]  
 dist = math.hypot(xr2-xr1,yr2-yr1)  
  
 if lndmrks[0] == 'Left':  
 setBrightness(dist)  
 elif lndmrks[0] == 'Right':  
 setVolume(dist)  
 elif lndmrks[0] == 'both':  
 xl1, yl1 = lndmrks[1][4][1], lndmrks[1][4][2]  
 xl2, yl2 = lndmrks[1][8][1], lndmrks[1][8][2]  
 distl = math.hypot(xl2 - xl1, yl2 - yl1)  
   
 t1 = threading.Thread(target=setVolume, args=(dist,))  
 t2 = threading.Thread(target=setBrightness, args=(distl,))  
   
 t1.start()  
 t2.start()  
  
   
  
 cv2.imshow("stream",frame)  
 if cv2.waitKey(1) & 0xFF == ord('q'):  
 break  
cv2.destroyAllWindows()

**CHAPTER 8**

**OUTPUT SCREEN**

**8.1 CONTROLLING VOLUME WITH LEFT HAND(MIN):**

When the thumb and index fingers distance is minimum the volume is minimum

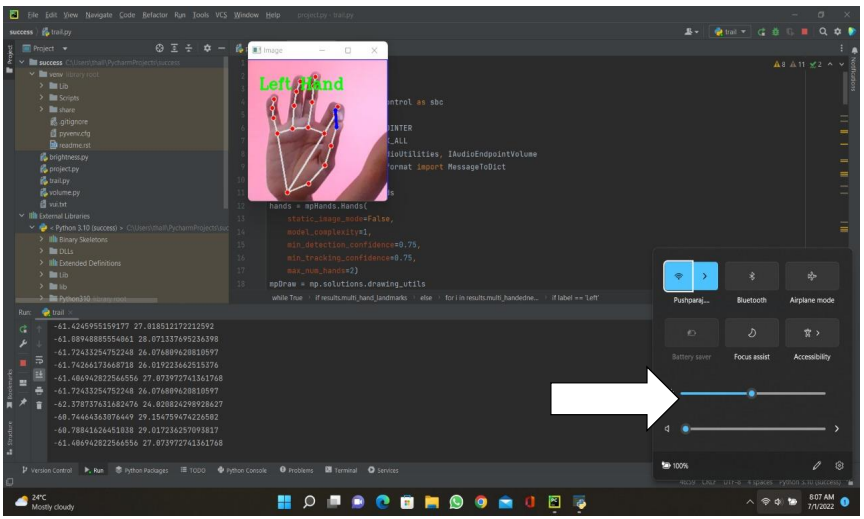
****

FIG-8.1 RIGHT HAND

**8.2 CONTROLLING VOLUME WITH LEFT HAND(MAX):**

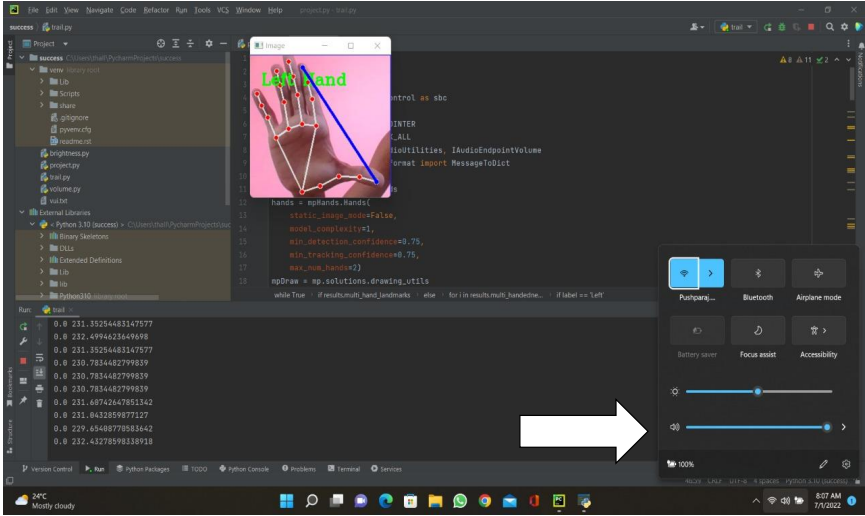
****

FIG-8.2 LEFT HAND

**8.3 CONTROLLING VOLUME WITH LEFT HAND:**

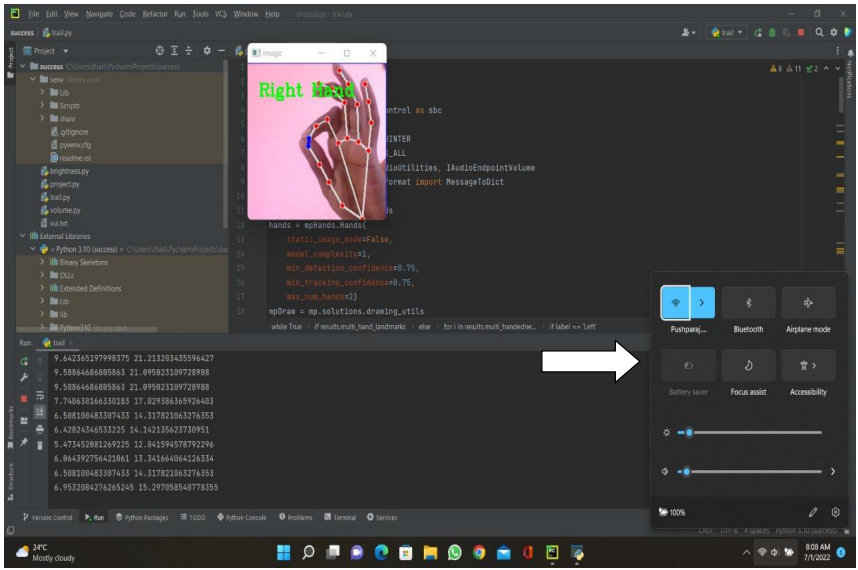
****

FIG-8.3 DATA COLLECTION PAGE

**CHAPTER 9**

**CONCLUSION**

The project presented a program that allowed user to perform hand gestures for

easy software control. A vision-based hand Gesture system that does not require

any special markers or gloves and can operate in real-time on a commodity PC with

low-cost cameras. Specifically, the system can track the tip positions of the counters

and index finger for each hand. The motivation for this hand Gesture was a desktop-

based

Hand gesture analysis can be divided into two main approaches, namely, glove-

based analysis, vision-based analysis . The glove-based approach employs sensors

(mechanical or optical) attached to a glove that acts as transducer of finger flexion

into electrical signals to determine hand posture.

**CHAPTER 10**

**FUTURE ENHANCEMENTS**

* In future we can use this feature TVs and Mobile phones also, Currently

its for PCs and Laptops.

* It is also going to help blind people to control volume by hand gestures.

**CHAPTER 11**

**REFERENCE**

* [**https://itsourcecode.com/free-projects/python-projects/brightness-control-withhand-detection-opencv-python-source-code/**](https://itsourcecode.com/free-projects/python-projects/brightness-control-withhand-detection-opencv-python-source-code/)
* [**https://github.com/i5han2/Volume\_and\_Brightness\_Control\_Using\_Hand\_Gestures**](https://github.com/i5han2/Volume_and_Brightness_Control_Using_Hand_Gestures)
* [**https://www.hackster.io/as4527/volume-control-using-hand-gesture-using-pythonand-opencv-7aab9f**](https://www.hackster.io/as4527/volume-control-using-hand-gesture-using-pythonand-opencv-7aab9f)
* [**https://www.section.io/engineering-education/creating-a-hand-gesture-volumecontroller-using-python-and-pycharm/**](https://www.section.io/engineering-education/creating-a-hand-gesture-volumecontroller-using-python-and-pycharm/)
* [**https://sourcecodehero.com/volume-control-with-hand-detection-opencv-pythonwith-source-code/**](https://sourcecodehero.com/volume-control-with-hand-detection-opencv-pythonwith-source-code/)