CONVERSION OF GESTURES TO VOICE AND TEXT MESSAGE IN REGIONAL LANGUAGE

A PROJECT REPORT

Submitted by

MENAKA S	**(211417104146)**
PREETHI P	**(211417104199)**
SHESHADRI ROSHNI SRUTHI	**(211417104257)**

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE ANDENGINEERING

PANIMALAR ENGINEERING COLLEGE, POONAMALLEE

ANNA UNIVERSITY: CHENNAI 600 025

AUGUST 2021

i

BONAFIDE CERTIFICATE

Certified that this project report **"CONVERSION OF GESTURES TO VOICE AND TEXT MESSAGE IN REGIONAL LANGUAGE"** is the bonafide work of "**MENAKA S (211417104146), PREETHI P (211417104199), SHESHADRI ROSHNI SRUTHI (211417104257)**)" who carried out the project work under my

SIGNATURE	**SIGNATURE**
Dr. S. MURUGAVALLI, M.E., Ph.D.,	**Mrs. V. SATHYA PREIYA., M.C.A., M.Phil., M.E.,**
HEAD OF DEPARTMENT	**SUPERVISOR**
DEPARTMENT OF CSE, PANIMALAR ENGINEERING COLLEGE, NAZARATHPETTAI, POONAMALLEE, CHENNAI-600 123.	DEPARTMENT OF CSE, PANIMALAR ENGINEERING COLLEGE, NAZARATHPETTAI, POONAMALLEE, CHENNAI-600 123

Certified that the above candidate(s) was/ were examined in the Anna University Project Viva-Voce Examination held on...**05/08/2021**…

INTERNAL EXAMINER   **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

<div align="right">

MENAKA S(211417104146)

PREETHI P (211417104199)

SHESHADRI ROSHNI SRUTHI(211417104257)

</div>

ABSTRACT

Speech and text is the main medium for human communication. A person needs vision to access the information in a text. However those who have poor vision can gather information from voice. This paper proposes a camera based assistive text reading to help visually impaired person in reading the text present on the captured image. The faces can also be detected when a person enter into the frame by the mode control. The proposed idea involves text extraction from scanned image using Tesseract Optical Character Recognition (OCR) and converting the text to speech by e-Speak tool, a process which makes visually impaired persons to read the text. This is a prototype for blind people to recognize the products in real world by extracting the text on image and converting it into speech. Computer vision is one of the emerging technologies that can be used to aid visually impaired people for navigation (both indoor and outdoor), accessing printed material, etc. This paper describes an approach to extract and recognize text from scene images effectively using computer vision technology and to convert recognized text into speech so that it can be incorporated with hardware to develop Electronic travel aid for visually impaired people in future.

TABLE OF CONTENT
LIST OF FIGURES

FIG NO   **TITLE**   **PAGE NO**

TABLE OF CONTENT

CHAPTER NO	TITLE	PAGE NO

CHAPTER 1
INTRODUCTION

CHAPTER 1
INTRODUCTION

1.1 OVERVIEW

Recent developments in computer software and related hardware technology have provided a value added service to the users. In everyday life, physical gestures are a powerful means of communication. They can economically convey a rich set of facts and feelings. For example, waving one's hand from side to side can mean anything from a "happy goodbye" to "caution". Use of the full potential of physical gesture is also something that most human computer dialogues lack.

The task of hand gesture recognition is one the important and elemental problem in computer vision. With recent advances in information technology and media, automated human interactions systems are build which involve hand processing task like hand detection, hand recognition and hand tracking.

This prompted the interest in making a software system that could recognize human gestures through computer vision, which is a sub field of artificial intelligence. The purpose of this software through computer vision was to program a computer to "understand" a scene or features in an image.

A first step in any hand processing system is to detect and localize hand in an image. The hand detection task was however challenging because of variability in the pose, orientation, location and scale. Also different lighting conditions add further variability.

1.2 PROBLEM DEFINITION

About billion people in the world are and dumb. The technology is developing day by day, but no significant development are undertaken for the betterment of deaf and dumb people. Communication between people have always been a challenging task, sign language helps deaf and dumb people to communicate. But not all people understand sign language. Gestures are considered as the most natural expressive way for communication between human and computer in virtual system. The main objective of this project is to achieve communication of deaf-mute people like a normal person. In recent year there is lot of research on gesture recognition using kinect sensor on using HD camera but camera and kinect sensors are more costly. This paper is focus on reduce cost and improve robustness of the proposed system using simple web camera.

CHAPTER 2
LITERATURE SURVEY

CHAPTER 2
LITERATURE SURVEY

Paper 1 - HAND GESTURE RECOGNITION AND VOICE CONVERSION FOR DEAF AND DUMB,

S. Vigneshwaran; M. Shifa Fathima ; V. Vijay Sagar ; R. SreeArshika,International Conference on Advanced Computing & Communication Systems (ICACCS)

Sign language plays a major role for dump people to communicate with normal people. It is very difficult for mute people to convey their message to normal people. Since normal people are not trained on hand sign language. In emergency time conveying their message is very difficult. So the solution for this problem is to convert the sign language into human hearing voice. There are two major techniques available to detect hand motion or gesture such as vision and non- vision technique and convert the detected information into voice through raspberry pi. In vision based technique camera will be used for gesture detection and non- vision based technique sensors are used. In this project non-vision based technique will be used. Most of the dumb people are deaf also. So the normal people's voice can be converted into their sign language. In an emergency situation the message will automatically send to their relation or friends.

Paper2 - SMARTGLOVES FOR HAND GESTURERE COGNITION USING SIGN LANGUAGE TO SPEECH CONVERSION SYSTEM,

International Conference on Robotics and Automation for Humanitarian Applications. Individuals with discourse impedance think that its hard to impart in a general public where the vast majority of the general population don't comprehend gesture based communication. The thought proposed in this paper is a keen glove which can change over communication via gestures to discourse yield. The glove is embedded with flex sensors and an mem sensor. A novel technique for State Estimation has been produced to track the movement of turn in three dimensional

spaces. The model was tried for its achievability in changing Indian Sign Language to voice yield. Despite the fact that the glove is expected for communication through signing to discourse transformation. mouth is established for the dumb people to overcome the complexity. It works on motion sensor, where the sensor reacts for every actions by the user. Database stores the messages and also all the templates. In the real time the template database is fed into a microcontroller and the motion sensor is fixed in their hand. For every action the motion sensors get accelerated and give the signal to the microcontroller. The microcontroller matches the motion with the database and produces the speech signal. The artificial mouth helps in retrieving the data from the database and the dumb people will speak like a normal person. The speakers are the output device wherein the system blocks which interprets the matched gestures as a text to speech conversion. The Embedded hardware with Flex sensor and micro sensors are placed which helps in reading the gestures performed by the user

Paper 3 - DESIGN AND IMPLEMENTATION OF A SIGN-TO-SPEECH/TEXT SYSTEM FOR DEAF AND DUMB PEOPLE,

Dalal Abdulla ; Shahrazad Abdulla ; Rameesa Manaf; Anwar H. Jarndal

This paper presents an approach for designing and implementing a smart glove for deaf and dumb people. There have been several researches done in order to find an easier way for non-vocal people to communicate with vocal people and express themselves to the hearing world. Developments have been made in sign language but mainly in American Sign Language. This research aims to develop a sign to Arabic language translator based on smart glove interfaced wirelessly with microcontroller and text/voice presenting devices. An approach has been developed and programmed to display Arabic text. The whole system has been implemented, programmed, cased and tested with very good results.

Paper 4 - **GESTURE AIDED SPEECH FOR DEAF AND MUTE,**

Dipti Jadhav ; Amiya Tripathy

Speech impaired people make use of sign language to communicate along with normal people. Common people also face difficulties to understand the gesture language. In order to minimize these real time issues, an attempt has been made to develop a system which consists of Flex sensors attached with the Data gloves along all fingers. These will be wearable gloves converts hand gestures into an audio output to interpret the expression. This reduces the communication gap that exists between mute and ordinary people. The proposed system plays corresponding recorded voice on an android phone connected to the hardware system via Bluetooth Module as an interpretation of the gesture. It is achieved by integrating flex sensor and 8051 microcontroller. The phone will use a text to speech converter to produce the output of corresponding gesture. This system offers high reliability and fast response as microcontroller works at a speed of 12 MHz. Processing time of microcontroller is much less than the time taken for a human to change from one gesture to another, the system works efficiently in real time scenario hence making the system faster.

Paper 5 - **IMPLEMENTATION OF GESTURE BASED VOICE AND LANGUAGE TRANSLATOR FOR DUMB PEOPLE,**

L. Anusha ; Y. Usha Devi ,International Conference on Communication and Electronics Systems (ICCES)

Dumb persons communicate through gestures which are not understood by the majority of people. Gesture is a movement of part of the body, especially a hand or the head, to express an idea or meaning. This paper proposes a system that converts gestures given by the user in the form of English alphabets into

corresponding voice and translates this English voice output into any other Microsoft supported languages. The system consists of MPU6050 for sensing gesture movement, Raspberry pi for processing, three button Keypad and speaker. It is implemented by using trajectory recognition algorithm for recognizing alphabets. Raspberry pi generates voice output for the text in multiple languages using voice RSS and Microsoft translator. When tested, the system recognized A-Z alphabets and generated voice output based on the gestures in multiple languages.

CHAPTER 3
SYSTEM ANALYSIS

CHAPTER 3
SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

In recent decades, due to computer software and hardware technologies of continuous innovation and breakthrough, the social life and information technology have a very close relationship in the twenty-first century. In the future, especially the interfaces of consumer electronics products (e.g. smart phones, games and infota1inment systems) will have more and more functions and be complex. How to develop a convenient human-machine Interface (Human Machine Interaction/Interface, HMI) for each consumer electronics product has become an important issue. The traditional electronic input devices, such as mouse, keyboard, and joystick are still the most common interaction way. However, it does not mean that these devices are the most convenient and natural input devices for most users. Since ancient times, gestures are a major way for communication and interaction between people.

People can easily express the idea by gestures before the invention of language. Nowadays, gestures still are naturally used by many people and especially are the most major and nature interaction way for deaf people. In recent years, the gesture control technique has become a new developmental trend for many human based electronics products, such as computers, televisions, and games. This technique let people can control these classifiers selection are a major issue in most researches. The third stage is to analyze sequential gestures to identify users' instructs or behaviors.

3.2 PROPOSED SYSTEM

Most gesture recognition methods usually contain three major stages. The first stage is the object detection. The target of this stage is to detect hand objects in the digital images or videos. Many environment and image problems are needed to solve at this stage to ensure that the hand contours or regions can be extracted precisely to enhance the recognition accuracy. Common image problems contain unstable brightness, noise, poor resolution and contrast. The better environment and camera devices can effectively improve these problems. However, it is hard to control when the gesture recognition system is working in the real environment or is become a product. Hence, the image processing method is a better solution to solve these image problems to construct an adaptive and robust gesture recognition system. The second stage is object recognition. The detected hand objects are recognized to identify the gestures. At this stage, differentiated features and effective.

Command line tools,[citation needed] or download MySQL front-ends from various parties that have developed desktop software and web applications to manage MySQL databases, build database structures, and work with data records.

THREE STAGES

(1) Hand detection

(2) Hand gesture recognition

(3) Finger detection

Now in this project by using part-based hand gesture recognition system. Which illustrates the framework, which consists of two major modules - hand detection and hand gesture recognition. Hand Detection - by using camera as the input device, which captures the color image and the depth map at 640*480 resolution. In order to segment the hand shape, and locate the hand position by

using the hand tracking function. Then, by thresholding from the hand position with a certain depth interval, a rough hand region can be obtained. Second, In require the user to wear a black belt on the gesturing hands wrist, in order to more accurately segment the problem as a distance measure, i.e., a measure of dissimilarity, one seeks the least costly transportation the movement of earth that requires the least amount of work. References and applied EMD to shape matching and contour retrieval, which represents the contour by a set of local descriptive features and computes the set of correspondences with minimum EMD costs between the local features. However, the existing EMD-based contour matching algorithms have two deficiencies when applied to hand gesture recognition - Two and shapes mainly in global features while not local features. The fingers (global features) are their major difference. Besides, the large number of local features slows down the speed of contour matching. Therefore, it is better to consider global features on matching. EMD allows for partial matching, i.e., a signature and its subset are considered to be the same in EMD measure. In this Finger-Earth Movers Distance (FEMD)can address these two deficiencies of the contour matching methods using EMD. Different from the EMD-based algorithm which considers each local feature as a cluster, by representing the input hand by global features (the finger clusters).And addition penalty on empty holes to alleviate partial matches on global features.

Finger Detection In order to measure the FEMD distance between hand shapes, need to represent the hand shape as a signature with each finger as a cluster, namely, to detect the finger parts from the hand shape. In this project two finger detection methods is proposed to obtain the finger parts from the hand

ADVANTAGES

❖ **Reduce external Interface** The Advantage of System is to Reduce External Interface like Mouse and Keyboard.

❖ **High Portability** The proposed System reduce the working of external interface

like keyboard and mouse so it makes it high portable.

3.3   REQUIREMENT ANALYSIS AND SPECIFICATION

3.3.1 HARDWARE REQUIREMENTS
- Desktop /Laptop
- GBRAM
- Keyboard
- Mouse
- Windows 7+
- Pentium IV+

3.3.2 SOFTWARE REQUIREMENTS
- Python
- IDLE
- Anaconda
- Jupyter Notebook

3.3.3 FUNCTIONAL REQUIREMENTS

Hand gesture recognition system can be divided into following modules -
- Preprocessing
- Feature extraction of the processed image
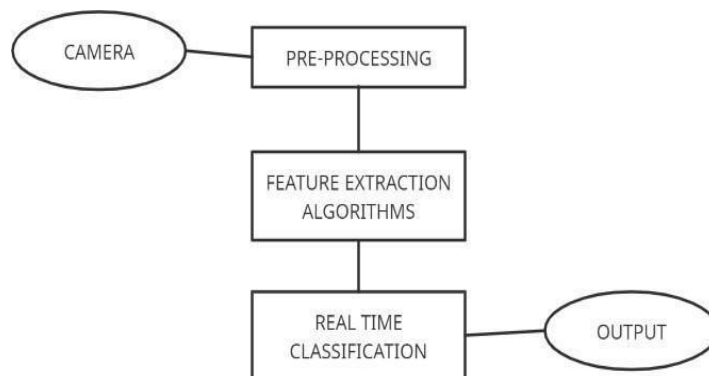 - Real time classification

Fig.no.3.3.1- **SYSTEM IMPLEMENTATION**

Value measure intensity or brightness. This is well enough to choose single color but it ignores complexity of color appearance. It trade off computation speed mean computationally expensive and perceptual relevance.

PRE-PROCESSING

Like many other pattern recognition tasks, pre-processing is necessary for enhancing robustness and recognition accuracy. The preprocessing prepares the image sequence for the recognition, so before calculating the diagonal Sum and other algorithms, a pre-processing step is performed to get the appropriate image, which is required for real time classification. So it consists of some steps. The net effect of this processing is to extract the hand only from the given input because once the hand is detected from the given input it can be recognized easily.

So pre-processing step mainly consists of following tasks -

- Skin Modeling
- Removal of Back ground
- Conversion from RGB to binary
- Hand Detection

SKIN MODELLING

      There are numerous method used for skin detection such as RGB (Red, Green, Blue), YCbCr (Luminance Chrominance) and HSV (Hue, Saturation, Value).

❖ **RGB**

RGB is a 3D color space pixel where each pixel has combination of three colors Red, Green and Blue at specific location. This technique widely used in image processing for identifying skin region.

❖ **YCBCR (LUMINANCE CHROMINANCE)**

This color space is used in digital video color information represent two color Cb and Cr. Cb is difference between Blue and Cr is difference between Red component references of value. This is basically RGB transformation to YCbCr for separation of luminance and chrominance for color modelling.

❖ **HSV (HUE, SATURATION AND VALUE)**

In HSV, Hue detect dominant color and Saturation define colorfulness whilst the approach for this thesis is to work with RGB to binarization techniques to Explicitly Defined skin Region.

❖ **SKIN DETECTION**

The skin color detection is one of important goal in hand gesture recognition. Skin color detection decision rules which have to be build will discriminate between skin portion and non-skin portion pixels. This is accomplished usually by metric introduction, which measure distance of the pixel color. This metric type is knows as skin modelling.

❖ **EXPLICITLY DEFINED SKIN REGION**

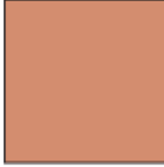Following are some common ethnic skin groups and there RGB color space:

European	Middle Eastern	Eastern	Asian	Lt. Black	Dk. Black
R=245	R=237	R=211	R=233	R=197	R=96
G=218	G=191	G=141	G=183	G=132	G=59
B=204	B=166	B=111	B=138	B=92	B=43

Fig.no.3.3.2 - Different Ethnic Group Skin Patches

To build a skin classifier is to define explicitly through a number of rules the boundaries of skin color cluster in some color space. The advantage of this method is the simplicity of skin detection rules that leads to the construction of very rapid classifier.

For Example (**R,G,B**) is classified as skin if-

R > 95 and **G >40** and **B > 20** and

max{R,G,B}−min{R,G,B}>15 and

|R−G| > 15 and **R > G** and **R > B**

In this classifier threshold defined to maximize the chance for recognizing the skin region for each color. In **Fig.no.3.3.2** that Red color in every skin sample is greater than 95, Green is greater than 40 and Blue is greater than 20. So threshold can make this classifier easily detect almost all kind of skin. This is one of the easiest methods as it explicitly defines skin-color boundaries in different color spaces. Different ranges of thresholds are defined according to each color space components in as the image pixels that fall between the predefined ranges are considered as skin pixels. The advantage of this method is obviously the simplicity which normally avoids of attempting too complex rules to prevent over

fitting data. However, it is important to select good color space and suitable decision rules to achieve high recognition rate with this method.

REMOVAL OF BACKGROUND

In identifying the background that greatly affects the results of hand detection decided to remove it. For this I have written own code in spite of using any built-in ones.

Before   **After**

Fig.no.3.3.3 - REMOVAL OF BACKGROUND

CONVERSION FROM RGB TO BINARY

All algorithms accept an input in RGB form and then convert it into binary format in order to provide ease in recognizing any gesture and also retaining the luminance factor in an image.

HAND DETECTION

Image could have more than one skin area but in requirement only hand for further process. For this i choose criteria image labeling which is following-

Labeling

To define how many skin regions that we have in image is by labelling all skin regions. Label is basically an integer value have 8 connecting objects in order to label all skin area pixel. If object had label then mark current pixel with label if not then use new label with new integer value. After counting all labelled region (segmented image) I sort all them into ascending order with maximum value and

24

choose the area have maximum value which I interested because I assume that hand region in bigger part of image. To separate that region which looked for, create new image that have one in positions where the label occurs and others set to zero.
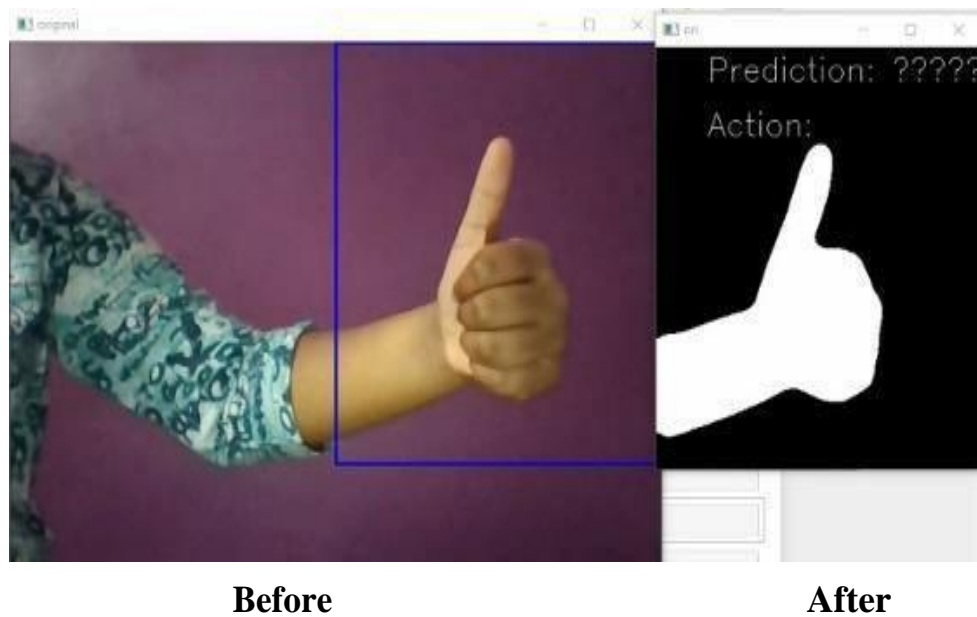

Before   **After**

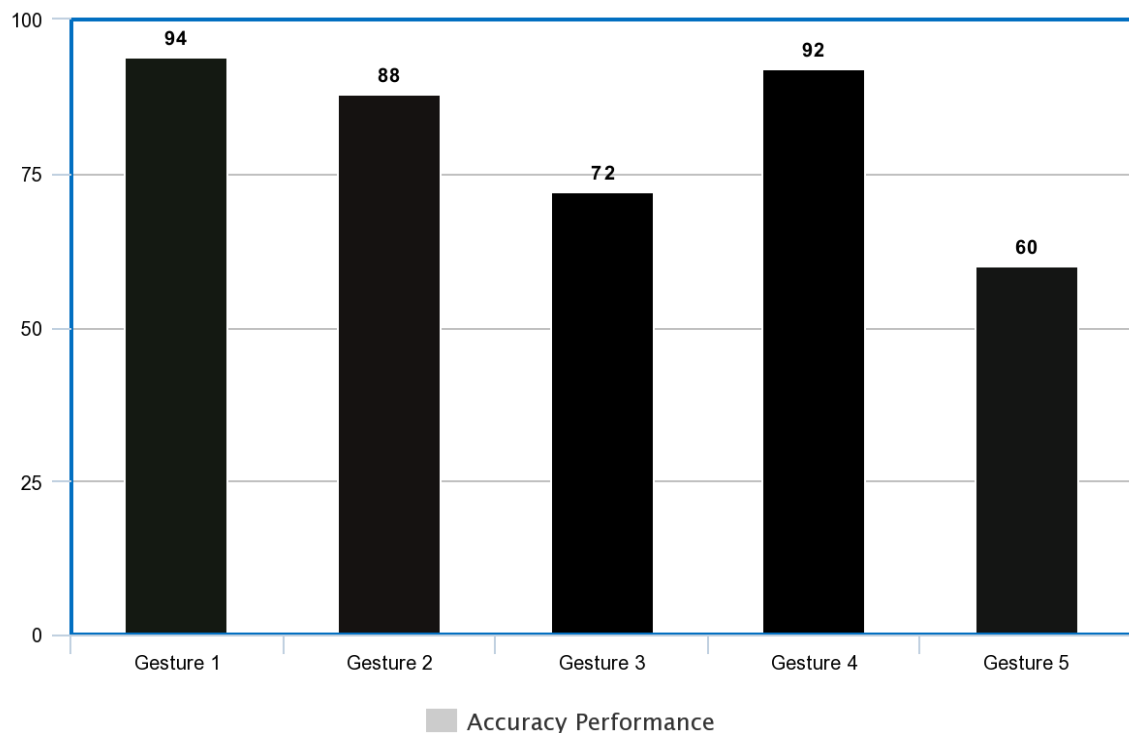**Fig.no.3.3.4 - LABELING SKIN REGION**

Fig.no.3.3.5 – ACCURACY PERFORMANCE

3.4 TECHNOLOGY STACK

PYTHON

Python is a general-purpose interpreted, interactive, object-oriented, and high- level programming language. It was created by Guido van Rossum during 1985- 1990. Like Perl, Python source code is also available under the General Public License (GPL). This tutorial gives enough understanding on Python programming language.

Python is a high-level, interpreted, interactive and object- oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

Python is Interpreted − Python is processed at runtime by the interpreter. No do not need to compile the program before executing it. This is similar to PERL and PHP.

Python is Object-Oriented − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

Python is a Beginner Language − Python is a great language for the beginner- level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

HISTORY OF PYTHON

❖ Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

❖ Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, Small Talk, and UNIX shell and other scripting languages.

❖ Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License(GPL).

❖ Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

PYTHON FEATURES

Python features include–

- **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.

- **Easy-to-read** − Python code is more clearly defined and visible to the yes.

- **Easy-to-maintain** − Python source code is fairly easy-to-maintain.

- **A broad standard library** − Python bulk of the library is very portable and

- cross-platform compatible on UNIX, Windows, and Macintosh.

- **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.

Portable − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.

Extendable –The low-level modules can be added to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.

Databases − Python provides interfaces to all major commercial databases.

GUI Programming − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

Scalable − Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below −

- ❖ It supports functional and structured programming methods as well as OOP.

- ❖ It can be used as a scripting language or can be compiled to byte-code for

- ❖ building large applications.
- ❖ It provides very high-level dynamic data types and supports dynamic type
- ❖ checking.
- ❖ It supports automatic garbage collection.
- ❖ It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

ANACONDA TOOL

Anaconda is a FREE enterprise-ready Python distribution for data analytics, processing, and scientific computing. Anaconda comes with Python 2.7 or Python 3.4 and 100+ cross-platforms tested and optimized Python packages. All of the usual Python ecosystem tools work with Anaconda. Additionally, Anaconda can create custom environments that mix and match different Python versions (2.6, 2.7, 3.3 or 3.4) and other packages into isolated environments and easily switch between them using conda, in this project with innovative multi-platform package manager for Python and other languages.

Anaconda Navigator

Anaconda Navigator is a desktop **graphical user interface (GUI)** included in Anaconda® distribution that allows to launch applications and easily manage conda packages, environments and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository.

It is available for Windows, macOS and Linux Using Python in Anaconda Many people write Python code using a text editor like Emacs or Vim. Others prefer to use an IDE like Spyder, Wing IDE, PyCharm or Python Tools for Visual Studio. Spyder is a great free IDE that is included with Anaconda. To start Spyder, type the name spyder in a terminal or at the Command Prompt.

The Python 2.7 version of Anaconda also includes a graphical Launcher application that enables to start IPython Notebook, IPythonQTConsole, and Spyder with a single click. On Mac, double click the Launcher.app, found in the

system ~/anaconda directory (or wherever the Anaconda installed). On Windows, on Start Menu there will be a Launcher. The Start Menu also has an Anaconda Command Prompt that, regardless of system and install settings, will launch the Python interpreter installed via Anaconda. This is particularly useful for troubleshooting, if the user have multiple Python installations on the system.

CONDA

❖   Conda is an open source package management system and environment management system that runs on Windows, macOS and Linux. Conda quickly installs, runs and updates packages and their dependencies. Conda easily creates, saves, loads and switches between environments on the local computer. It was created for Python programs, but it can package and distribute software for any language.

❖   Conda as a package manager helps to find and install packages. If in need a package that requires a different version of Python, do not need to switch to a different environment manager, because conda is also an environment manager. With just a few commands, it can set up a totally separate environment to run that different version of Python, while continuing to run the usual version of Python in user system normal environment.

SPYDER

❖   Spyder's text editor is a multi-language editor with features such as syntax coloring, code analysis (real-time code analysis powered by pyflakes and advanced code analysis using pylint), introspection capabilities such as code completion, calltips and go-to-definition features (powered by rope), function/class browser, horizontal/vertical splitting features, etc.Spyder is the Scientific Python Development Environment.

❖   Spyder is a powerful interactive development environment for the Pythonlanguagewithadvancedediting,interactivetesting,debuggingand

Introspection features; and a numerical computing environment thanks to the support of IPython (enhanced interactive Python interpreter) and popular Python libraries such as NumPy (linear algebra), SciPy (signal and image processing) ormatplotlib (interactive 2D/3Dplotting).

JUPYTER NOTEBOOK

❖ Notebook documents (or "notebooks", all lower case) are documents produced by the Jupyter Notebook App, which contain both computer code (e.g. python)and rich text elements (paragraph, equations, figures, links, etc…). Notebook documents are both human-readable documents containing the analysis description and the results (figures, tables, etc..) as well as executable documents which can be run to perform data analysis.

JUPYTER NOTEBOOK APP

❖ The Jupyter Notebook App is a server-client application that allows editing and running notebook documents via a web browser. The Jupyter Notebook App can be executed on a local desktop requiring no internet access (as described in this document) or can be installed on a remote server and accessed through the internet.

❖ In addition to displaying/editing/running notebook documents, the Jupyter Notebook App has a "Dashboard" (Notebook Dashboard), a "control panel" showing local files and allowing to open notebook documents or shutting down their kernels.

KERNEL

❖ A notebook kernel is a "computational engine" that executes the code contained in a Notebook document. The ipython kernel, referenced in this guide, executes python code.

❖ Kernels for many other languages exist (official kernels).When Notebook

document is opened, the associated kernel is automatically launched. When the notebook is executed (either cell-by-cell or with menu Cell Depending on the type of computations, the kernel may consume significant CPU and RAM. Note that the RAM is not released until the kernel is shut-down Notebook Dashboard The Notebook Dashboard is the component which is shown first when Jupyter Notebook App launched.

❖ The Notebook Dashboard is mainly used to open notebook documents, and
to manage the running kernels (visualize and shutdown).

❖ The Notebook Dashboard has other features similar to a file manager, namely

❖ navigating folders and renaming/deleting files.

In this case, "notebook" or "notebookdocuments" denote documents that contain both code and rich text elements, such as figures, links, and equations. Because of the mix of code and text elements, these documents are the ideal place to bring together an analysis description and its results as well as they can be executed perform the data analysis in real time. These documents are produced by the Jupyter Notebook App.

For now, it is should be know that "Jupyter" is a loose acronym meaning Julia, Python, and R. These programming languages were the first target languages of the Jupyter application, but nowadays, the notebook technology also supports many other languages.

CHAPTER 4
SYSTEM DESIGN

4.1 UML DIAGRAMS

Unified Modeling Language (UML) is a general purpose modelling 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.

4.1.1 USE CASE DIAGRAM

A use case diagram at its simplest is a representation of a user's interaction
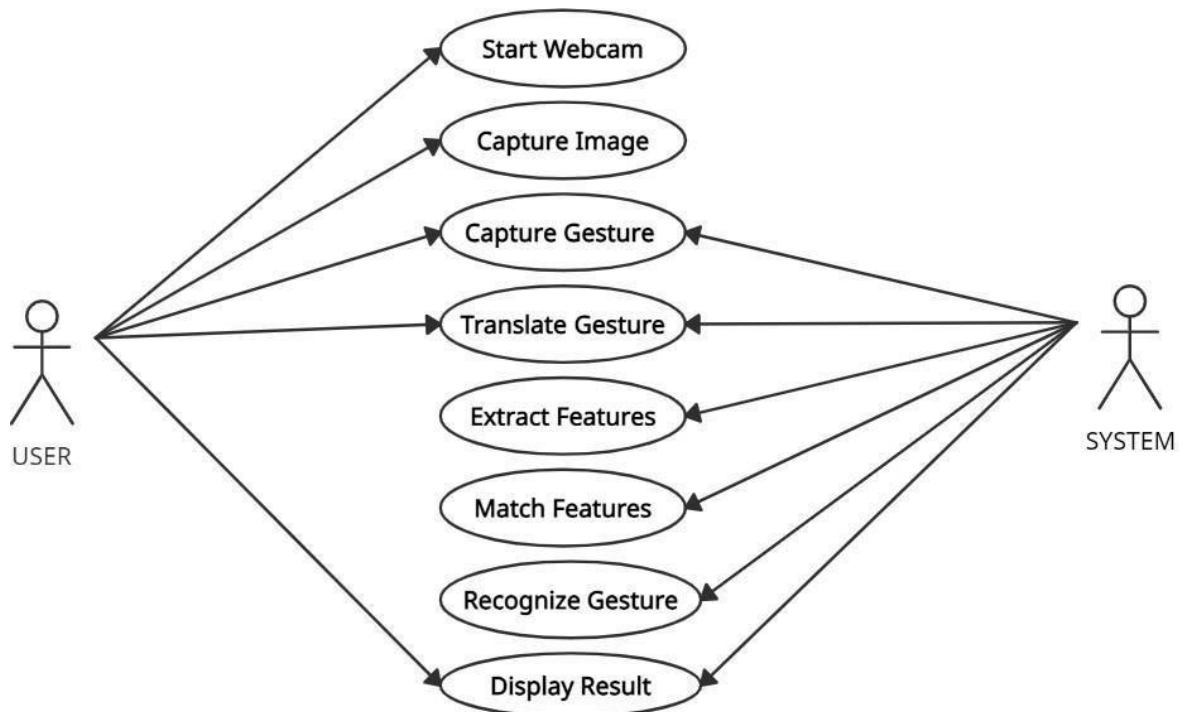
with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well.
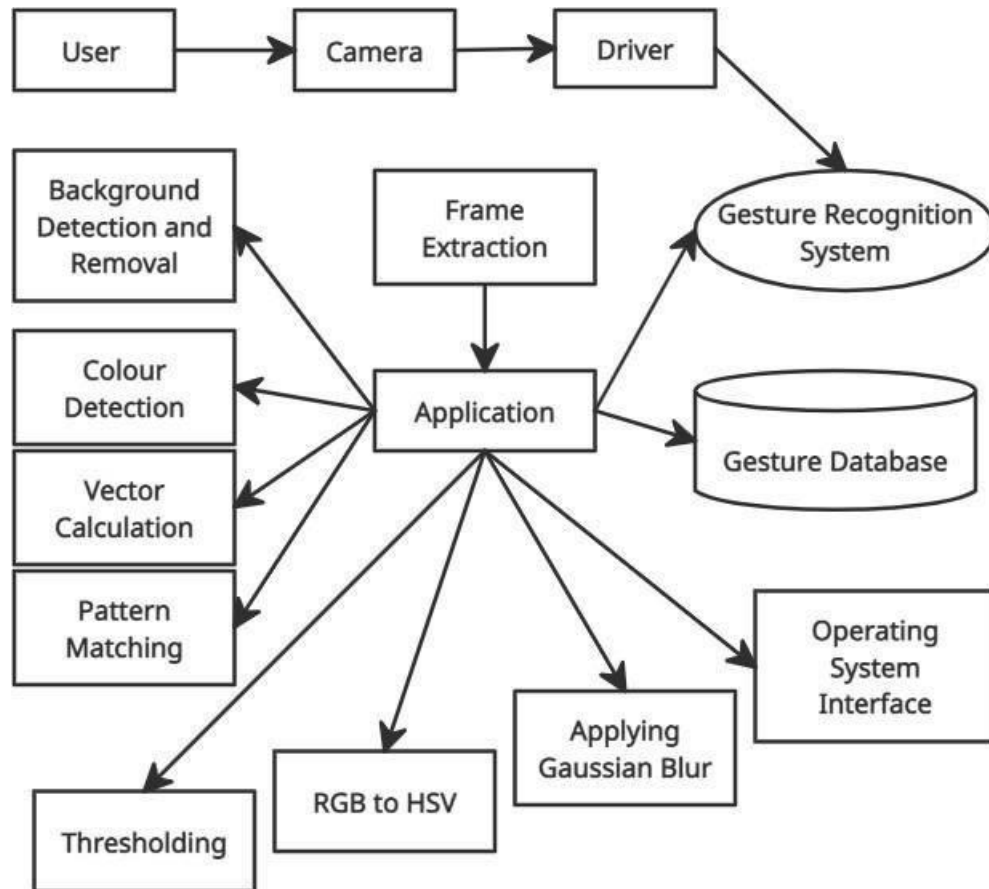
4.3.2 DATA FLOW DIAGRAM

Fig.no.4.1.2 - DATA FLOW DIAGRAM

4.1.3 FLOW CHART DIAGRAM

A diagram of the sequence of movements or actions of people or things involved in a complex system or activity.
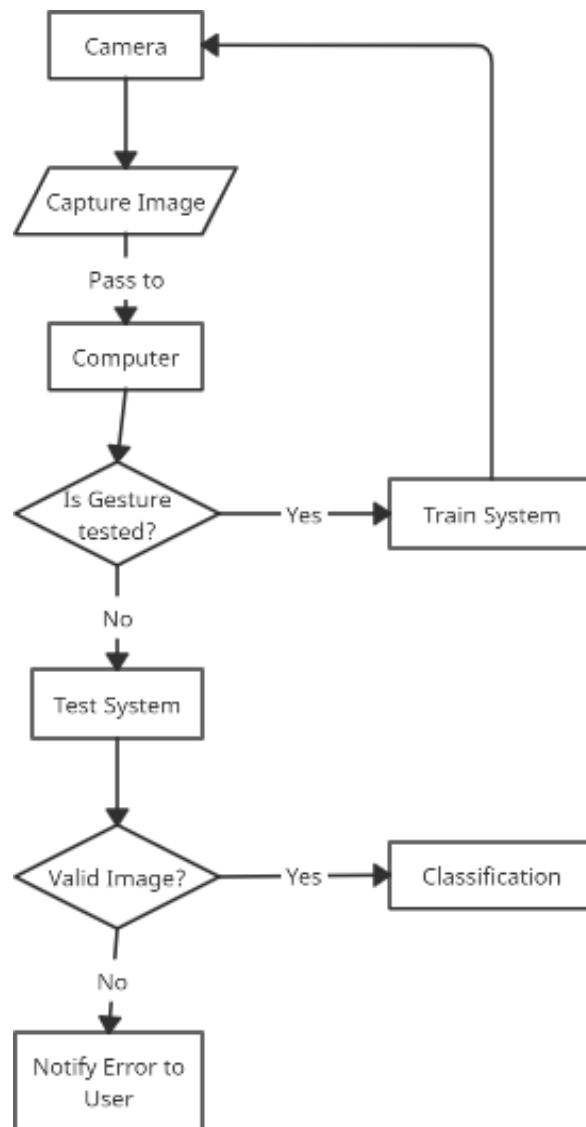

Fig.no.4.1.3 - FLOW CHART DIAGRAM

4.1.4 ACTIVITY DIAGRAM

A graphical representation of an executed set of procedural system activities and considered a state chart diagram variation. Activity diagrams describe parallel and conditional activities, use cases and system functions at a detailed level.
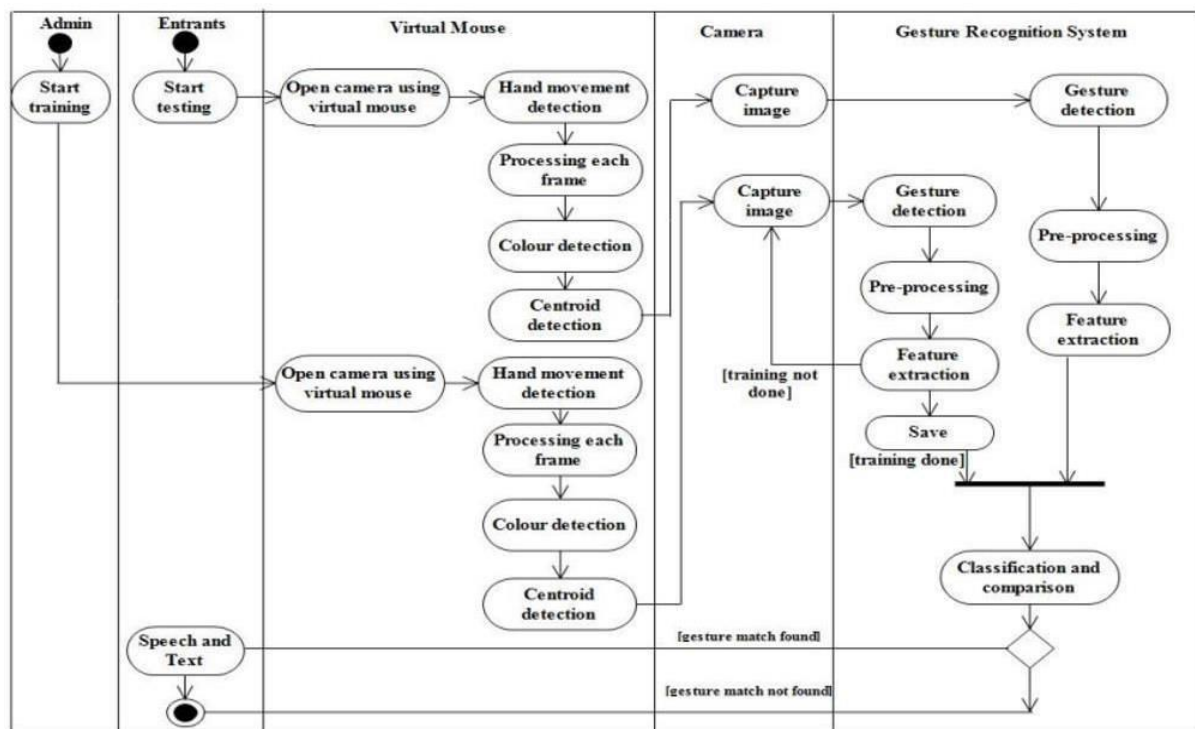

Fig.no.4.1.4 - ACTIVITY DIAGRAM

CHAPTER 5
SYSTEM ARCHITECTURE

CHAPTER 5

SYSTEM
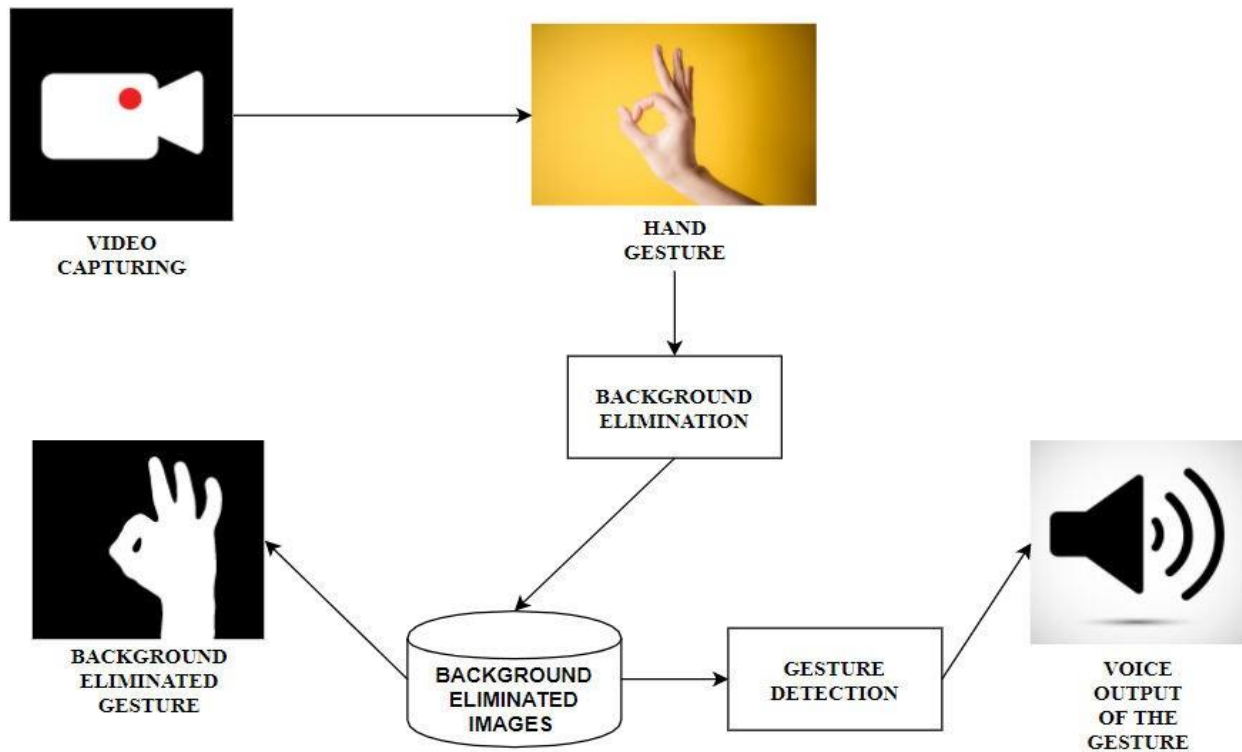
ARCHITECTURE

5.1 ARCHITECTURE OVERVIEW

Fig.no.5.1 ARCHITECTURE DIAGRAM

The figure above represents the System Architecture of the project that basically show each component of the system , how the system works, and the flow of the system and so on. Images that are taken from the web camera goes under preprocessing stages to enhance the feature of an image. Then there is a removal of object and background from the images which later convert into binary form. Feature extraction and reorganization helps to match the images that is stored in database and get the desired output in the form of text and converts that text to speech.

5.2 MODULES DESIGN SPECIFICATION

❖ Dataset collection

❖ Dataset pre-processing

❖ Machine Learning Training

❖ Capture Real time video

❖ Eliminate Background

❖ Identify Gesture

Dataset Collection

❖ Dataset collection involves collecting or creating test dataset for gesture and respective terminology. This is very exhaustive task as in need to collect more samples for gesture. Dataset are collected from kaggle website for the project in scope.

Dataset Pre-Processing

❖ Dataset preprocessing involves removing noise in images and converting to numpy arrays

Machine Learning

❖ Machine learning involves training the system and classification using CNN algorithms.

Capture Real Time Video & Gesture Detection

❖ By using the open cv to capture and process the video. After capturing video then pre-process the frames and fit it in model to get the gesture right. **Background Elimination**

❖ And by finding the contours to detect edges of the main object and create a mask with numpy zeros for the background and then combine the mask and the image using bitwise and operator.

5.2 PROGRAM DESIGN LANGUAGE

ALGORITHM

Step 1 - Reading image from camera and applying pre-processing techniques like gamma correction, blurring.

Step 2 - Hand Segmentation using background subtraction

algorithm. Step 3 - Hand detection using thresholding and dilation.

Step 4 - Finding contours of hand for getting shape of hand.

Step 5 - Finding contour area, convex hull, hull area,

solidity.

Step 6 - Also find the angle between two fingers and aspect ratio of hand.

Step 7 - Finding the defects of hand using convex hull.

Step 8 - Finally classifying using solidity, aspect ratio, convex defects and

angle. Step 9 - if image (sign) ==database image, speak the meaning of that

sign Repeat. Step 10 - END.

CHAPTER 6 SYSTEM IMPLEMENTATION

1. SYSTEM IMPLEMENTATION

1.1 SYSTEM TRAIN CODING

IN [1]   import

osimport

warnings import

cv2 import keras

    import matplotlib.pyplot as plt

    import matplotlib.style as style

    import numpy as np

    import pandas as pd

    from PIL import Image

    from keras import models, layers, optimizers

    from keras.applications import VGG16

    from keras.callbacks import EarlyStopping, ModelCheckpoint

    from keras.layers import Dense, Dropout, Flatten

    from keras.models import Model

    from keras.preprocessing import image as image_utils

    from keras.preprocessing.image import ImageDataGenerator

    from keras.utils import to_categorical

    from sklearn.metrics import classification_report, confusion_matrix

    from sklearn.model_selection import train_test_split

    #% matplotlib inline

    style.use('seaborn-whitegrid')

    warnings.filterwarnings(action='once')

IN [2] gestures = {'L_': 'L','fi': 'Fist','C_': 'C','ok': 'Okay', 'pe': 'Peace','pa': 'Palm' }

```python
gestures_map = {'Fist' : 0, 'L': 1,'Okay': 2,'Palm': 3,'Peace': 4}
```

IN[3]
```python
def process_image(path):
    img =Image.open(path)
    img = img.resize((224, 224)) img
    = np.array(img)
    return img


def process_data(X_data, y_data):
    X_data = np.array(X_data, dtype = 'float32') if
      rgb:
        pass else:
    X_data = np.stack((X_data,)*3, axis=-1)
    X_data /= 255
    y_data = np.array(y_data) y_data
    = to_categorical(y_data)
      return X_data, y_data


    def walk_file_tree(relative_path): X_data
    = []
    y_data = []
      for directory, subdirectories, files in os.walk(relative_path): for
        file in files:
          if not file.startswith('.') and (not file.startswith('C_')): path
            = os.path.join(directory, file) print('gestures',gestures)
    gesture_name = gestures[file[0:2]]
    print('gesture name',gesture_name)

  y_data.append(gestures_map[gesture_name])
  X_data.append(process_image(path))
```

```python
            else:
                continue
        X_data, y_data = process_data(X_data, y_data)
        return X_data, y_data
```

IN[4]   classData(object):
```python
    def __init__(self):
        self.X_data = []
        self.y_data = []
    def get_data(self):
        return self.X_data, self.y_data
```

IN [5] relative_path = 'C:\\Users\\User\\Desktop\\Gesture\\Project\\tr'
rgb = False
```python
    # # This method processes the data
    X_data, y_data = walk_file_tree(relative_path)
```
Can also optionally use a class to get this data, in order to keep itseparate from
Drawing data
```python
    silhouette = Data()
    silhouette.X_data, silhouette.y_data = walk_file_tree(relative_path)
```

IN [6]   print(f'X_data shape: {X_data.shape}')
print(f'y_data   shape:{y_data.shape}')

IN [7]plt.imshow(X_data[0])

IN [8] X_train, X_test, y_train, y_test = train_test_split(X_data, y_data, test_size =

```
0.2, random_state=12, stratify=y_data)

IN [9] file_path = 'saved_model.hdf5'
model_checkpoint = ModelCheckpoint(filepath=file_path,
save_best_only=True)
early_stopping = EarlyStopping(monitor='val_acc',
min_delta=0, patience=10, verbose=1, mode='auto',
    restore_best_weights=True)

IN [10] imageSize = 224
vgg_base = VGG16(weights='imagenet', include_top=False,
input_shape=(imageSize, imageSize, 3))
optimizer1 = optimizers.Adam()
base_model = vgg_base # Topless
# Add top layer
x = base_model.output
        x = Flatten()(x)
        x = Dense(128, activation='relu', name='fc1')(x)
        x = Dense(128, activation='relu', name='fc2')(x)
        x = Dense(128, activation='relu', name='fc3')(x)
        x = Dropout(0.5)(x)
        x = Dense(64, activation='relu', name='fc4')(x) predictions
        = Dense(5, activation='softmax')(x)
        model = Model(inputs=base_model.input, outputs=predictions)
        # Train top layers only
        for layer in base_model.layers: layer.trainable
        = False
        callbacks_list = [keras.callbacks.EarlyStopping(monitor='val_acc',
        patience=3, verbose=1)]
```

```python
model.compile(optimizer='Adam',     loss='categorical_crossentropy',
metrics=['accuracy'])
model.fit(X_train, y_train, epochs=5, batch_size=10,
validation_data=(X_train, y_train), verbose=1,
callbacks=[early_stopping,    model_checkpoint])
```

IN [11] model.save('gestmodel.h5')

IN [12] fromkeras.models import load_model
model = load_model('gestmodel.h5')

IN [13]   def   get_classification_metrics(X_test,y_test):
pred = model.predict(X_test)

```python
        pred = np.argmax(pred, axis=1)
        y_true = np.argmax(y_test, axis=1)
        print(confusion_matrix(y_true,pred))
        print('\n')
        print(classification_report(y_true,pred))
```

IN [14]   gesture_names = {0: 'Fist',1: 'L',2: 'Okay',3: 'Palm',4: 'Peace'}

```python
        defpredict_rgb_image(path):
            img2rgb = image_utils.load_img(path=path, target_size=(224, 224))
            img2rgb = image_utils.img_to_array(img2rgb)
            img2rgb = img2rgb.reshape(1, 224, 224, 3)
            return    gesture_names[np.argmax(model.predict(img2rgb))]
```

IN [15]   from PIL import Image

```python
        predict_rgb_image('C:\\Users\\User\\Desktop\\Gesture\\Project\\tr\\okay
_004.jpg')
```

6.2 CLIENT SIDE CODING

```python
#! /usr/bin/env python3
import cv2
import copy
import numpy as np
from keras.models import load_model
#from phue import Bridge
#fromsoco import SoCo
#import pygame
import time
importgoogletrans
from googletrans import Translator
import gtts as gt
import os
translator = Translator()


# General Settings
prediction = '94'
action = ''
score = 0
img_counter = 500
save_images, selected_gesture = True, 'peace'
gesture_names = {0: 'Fist',1: 'L',2: 'Okay', 3: 'Palm',4: 'Peace'}
model = load_model('gestmodel.h5')
def predict_rgb_image(img):
    result = gesture_names[model.predict_classes(img)[0]]
    print(result)
    return (result)
def predict_rgb_image_vgg(image):
    image = np.array(image, dtype='float32')
```

```python
        image /= 255
pred_array = model.predict(image)
print(f'pred_array: {pred_array}')
    result = gesture_names[np.argmax(pred_array)]
print(f'Result: {result}')
    print(max(pred_array[0]))
    score = float("%0.2f" % (max(pred_array[0]) * 100))
    print(result)
    result = translator.translate(result,src='en',dest='ta')
    print(result.text)
tts=gt.gTTS(text=result.text,lang='ta')
tts.save("tamil.mp3")
os.system("tamil.mp3")
    return result.text, score
    cap_region_x_begin = 0.5 # start point/total width
    cap_region_y_end = 0.8 # start point/total width
    threshold = 60 # binary threshold
    blurValue = 41 #GaussianBlur parameter
    bgSubThreshold = 50
    learningRate = 0

    # variableslt
    isBgCaptured = 0 # bool, whether the background captured
    triggerSwitch = False # if true, keyboard simulator works
def remove_background(frame):
    fgmask = bgModel.apply(frame, learningRate=learningRate)
        kernel = np.ones((3, 3), np.uint8)
    fgmask = cv2.erode(fgmask, kernel, iterations=1)
        res = cv2.bitwise_and(frame, frame, mask=fgmask)
```

```python
    returnres


        # Camera
        camera = cv2.VideoCapture(0)
        camera.set(10, 200)
        while camera.isOpened():
            ret, frame = camera.read()
            frame = cv2.bilateralFilter(frame, 5, 50, 100) # smoothing filter
            frame = cv2.flip(frame, 1) # flip the frame horizontally
            cv2.rectangle(frame, (int(cap_region_x_begin * frame.shape[1]), 0),
                    (frame.shape[1], int(cap_region_y_end * frame.shape[0])),
    (255, 0, 0), 2)
         cv2.imshow('original', frame)
         # Run once background is captured
            if isBgCaptured == 1:
        img = remove_background(frame)
        img = img[0:int(cap_region_y_end * frame.shape[0]),
    int(cap_region_x_begin * frame.shape[1]):frame.shape[1]]
        # clipthe ROI
            # convert the image into binary image
            gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
            blur = cv2.GaussianBlur(gray, (blurValue, blurValue), 0)
    ret, thresh = cv2.threshold(blur, threshold, 255,
    cv2.THRESH_BINARY+    cv2.THRESH_OTSU)


    cv2.putText(thresh, f"Prediction: {prediction} ({score}%)", (50, 30),
    cv2.FONT_HERSHEY_SIMPLEX, 1,(255, 255, 255))


            #print('length :' + len(action))
```

```python
        #if(len(action)>0):
cv2.putText(thresh, f"Action: {action}", (50, 80),
cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255))
# Draw the text
cv2.imshow('ori', thresh)
k = cv2.waitKey(10)
if k == 27: # press ESC to exit all windows at any time
break
        elif k == ord('b'): # press 'b' to capture the background
        bgModel = cv2.createBackgroundSubtractorMOG2(0,
        bgSubThreshold)
            #b.set_light(6, on_command)
        time.sleep(2)
        isBgCaptured        =        1
        print('Background captured')
            elif k == ord('r'): # press 'r' to reset the background
            time.sleep(1)
            bgModel = None
          triggerSwitch = False
          isBgCaptured = 0
          print('Reset background')
          elif k == 32:
              # If space bar pressed
              cv2.imshow('original', frame)
              # copies 1 channel BW image to all 3 RGB channels
              target = np.stack((thresh,) * 3, axis=-1)
              target = cv2.resize(target, (224,224))
              target = target.reshape(1, 224, 224,3)
              prediction, score = predict_rgb_image_vgg(target)
```

```python
 if save_images:
     cv2.imwrite('test.jpg',thresh)
     img_counter +=
     1 elif k ==
     ord('t'):
     print('Tracker turned on.')
         cap =
         cv2.VideoCapture(0)
         ret, frame =cap.read()
     # Select Region of Interest
     (ROI) r =
     cv2.selectROI(frame)
     # Crop image
imCrop = frame[int(r[1]):int(r[1] + r[3]), int(r[0]):int(r[0] + r[2])]
     # setup initial location of
     window r, h, c, w = 250,
     400, 400, 400
     track_window = (c, r, w, h)
     # set up the ROI for
     tracking roi =imCrop
     hsv_roi = cv2.cvtColor(roi, cv2.COLOR_BGR2HSV)
         mask = cv2.inRange(hsv_roi, np.array((0., 60., 32.)),
     np.array((180.,  255., 255.)))
     roi_hist = cv2.calcHist([hsv_roi], [0], mask, [180], [0, 180])
         cv2.normalize(roi_hist, roi_hist, 0, 255, cv2.NORM_MINMAX)
     # Setup the termination criteria, either 10 iteration or move by at
     least 1  pt
     term_crit = (cv2.TERM_CRITERIA_EPS |
     cv2.TERM_CRITERIA_COUNT, 10, 1)
```

```python
    while (1):
  ret, frame = cap.read()
      if ret == True:
   hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

dst = cv2.calcBackProject([hsv], [0], roi_hist, [0, 180], 1)
        # apply meanshift to get the new location
        ret, track_window = cv2.CamShift(dst, track_window,
term_crit)
        # Draw it on image
        pts = cv2.boxPoints(ret)
        pts = np.int0(pts)
        img2 = cv2.polylines(frame, [pts], True, (0, 255, 0), 2)
        cv2.imshow('img2', img2)
        k = cv2.waitKey(60) & 0xff
        if k == 27: # if ESC key
            break
        else:
            cv2.imwrite(chr(k) + ".jpg", img2)

            else:
                break
            cv2.destroyAllWin
            dows()
        cap.release()
```

CHAPTER 7

SYSTEM TESTING

7.1 TEST CASES & REPORTS / PERFORMANCE ANALYSIS
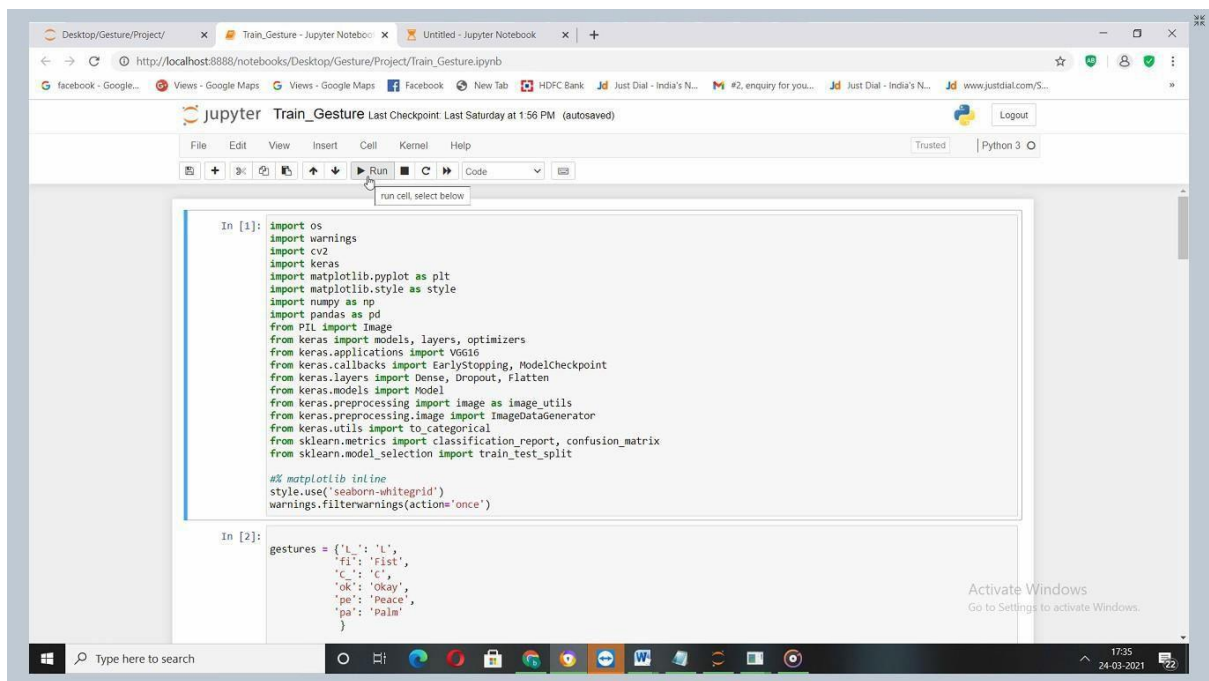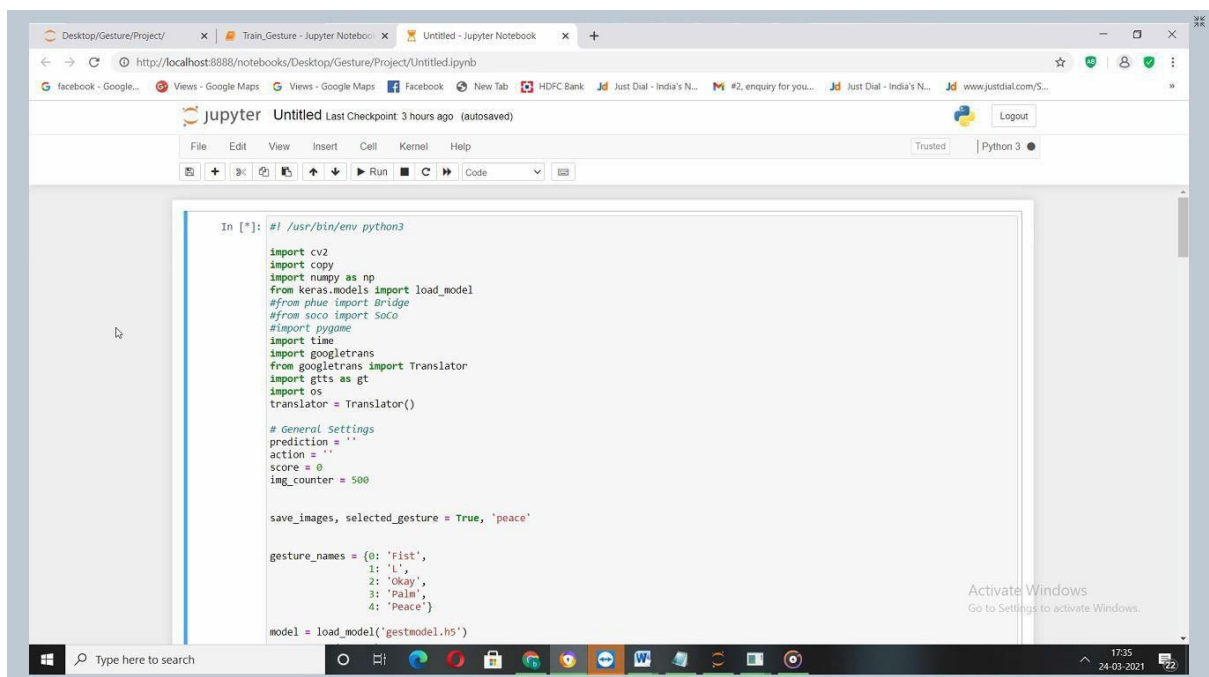

Fig.no.7.3.1 - System Training Coding

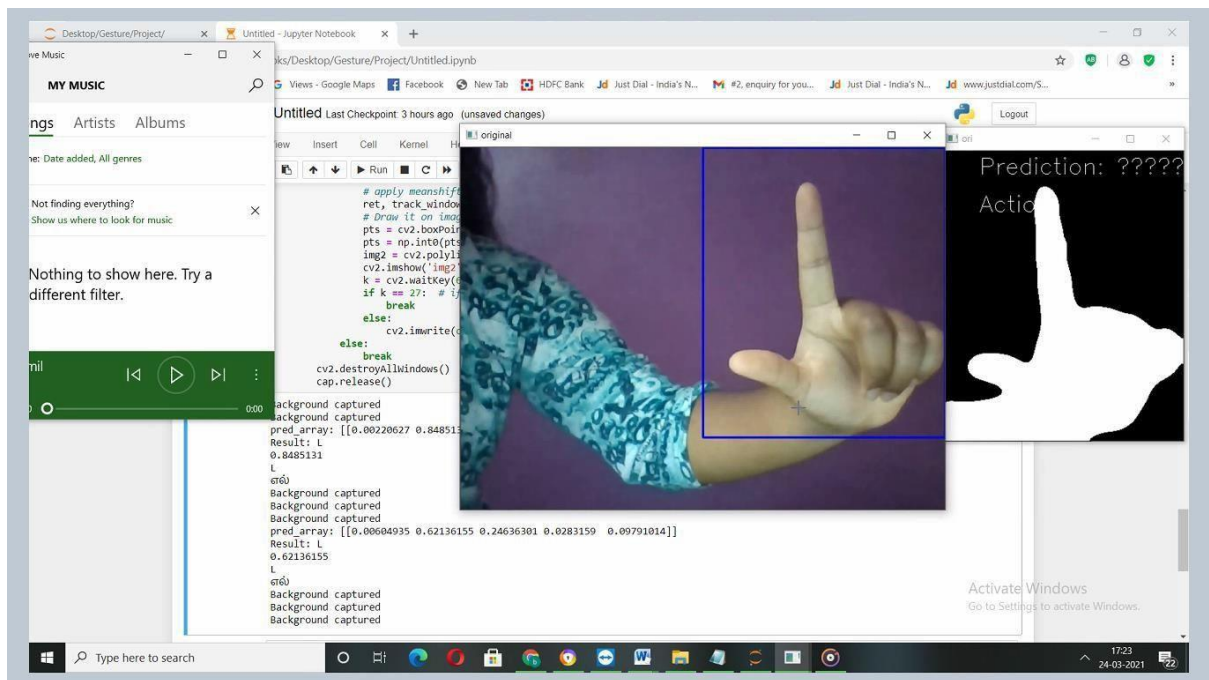Fig.no.7.3.2 - Client-Side Coding

Fig.no.7.3.3 - Sample Output Screen Shot

7.2 UNIT TESTING

Unit testing is a type of software testing where individual units or components of a software are tested. The purpose is to validate that each unit of the software code performs as expected. Unit Testing is done during the development (coding phase) of an application by the developers. Unit Tests isolate a section of code and verify its correctness. A unit may be an individual function, method, procedure, module, or object. In SDLC, STLC, V Model, Unit testing is first level of testing done before integration testing. Unit testing is a White Box testing technique that is usually performed by the developer. Though, in a practical world due to time crunch or reluctance of developers to tests, QA engineers also do unit testing.

7.3 INTEGRATION TESTING

Integration testing is a technique for constructing the software architecture and conducting tests to uncover errors with interface. The objective of testing was to crosscheck for components fully functional or not according to design. Thus, by integrating all the unit components and if the system worked as a whole properly or not. The information flows between the components were checked once again.

CHAPTER 8
CONCLUSION

CHAPTER 8
CONCLUSION

8.1 CONCLUSION AND FUTURE ENHANCEMENTS

8.1.1 CONCLUSION

This chapter summarizes my work at every stage of the project. At the time I started my thesis, I had a brief idea of how I will bring it from a topic on the paper to a real product. Due to knowledge of Computer Vision and Biometric subjects I had background in the image-processing field but not at expert level but my constant effort helped me to go through and succeed eventually.

As required in every project, research is of utmost importance. So, by spending pretty much time in going through the background literature. And looked at various approaches of doing this thesis and developed four different methods: Row vector algorithm, Edging and row vector passing algorithm, Mean and standard deviation of edged image and Diagonal sum algorithm.

Each of these algorithms was tried with neural networks and have higher performance rate in the ascending order respectively.

The first limitation that was discovered in all the algorithms used with neural networks was that their performance depended on the amount of training dataset provided. The system worked efficiently after being trained by a larger dataset as compared to a smaller dataset.

The Row vector algorithm used initially was a very vague approach adopted for classification as it was found through experiments that the row vectors of two different images could happen to be the same.

In the edging and row vector-passing algorithm, the edging parameter was

introduced in addition to the row vector to improve the gesture classification

5

accuracy but it was found that due to self-shadowing effect found in edges, the detection rate was not sufficiently improved.

The next parameters tried for classification were mean and standard deviation. They also failed to give satisfactory results (i.e. above 60%) but still they were among the best parameters used for detection with neural networks.

Due to the unusual behavior of neural network with all the mentioned parameters, the diagonal sum algorithm was finally implemented in real time. The system was tested with 60 pictures and the detection rate was found to be 86%. The strengths and weaknesses of gesture recognition using diagonal sum have been presented and discussed. With the implemented system serving as an extendible foundation for future research, extensions to the current system have been proposed.

8.1.2 FUTURE ENHANCEMENTS

The system could also be made smart to be trained for only one or two gestures rather than all and then made ready for testing. This will require only a few changes in the current interface code, which were not performed due to the shortage of time.

One time training constraint for real time system can be removed if the algorithm is made efficient to work with all skin types and light conditions which seems impossible by now altogether. Framing with COG (Centre of gravity) to control orientation factor could make this system more perfect for real application.

The system's speed for preprocessing could be improved if the code is developed in VC/VC.Net
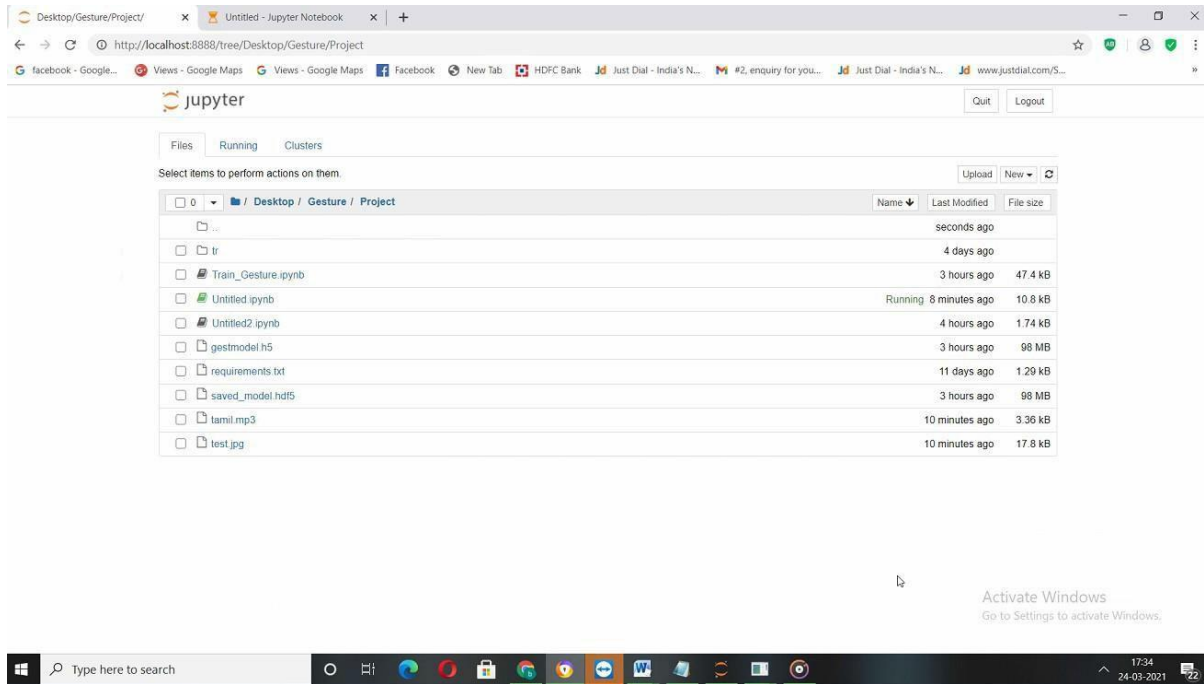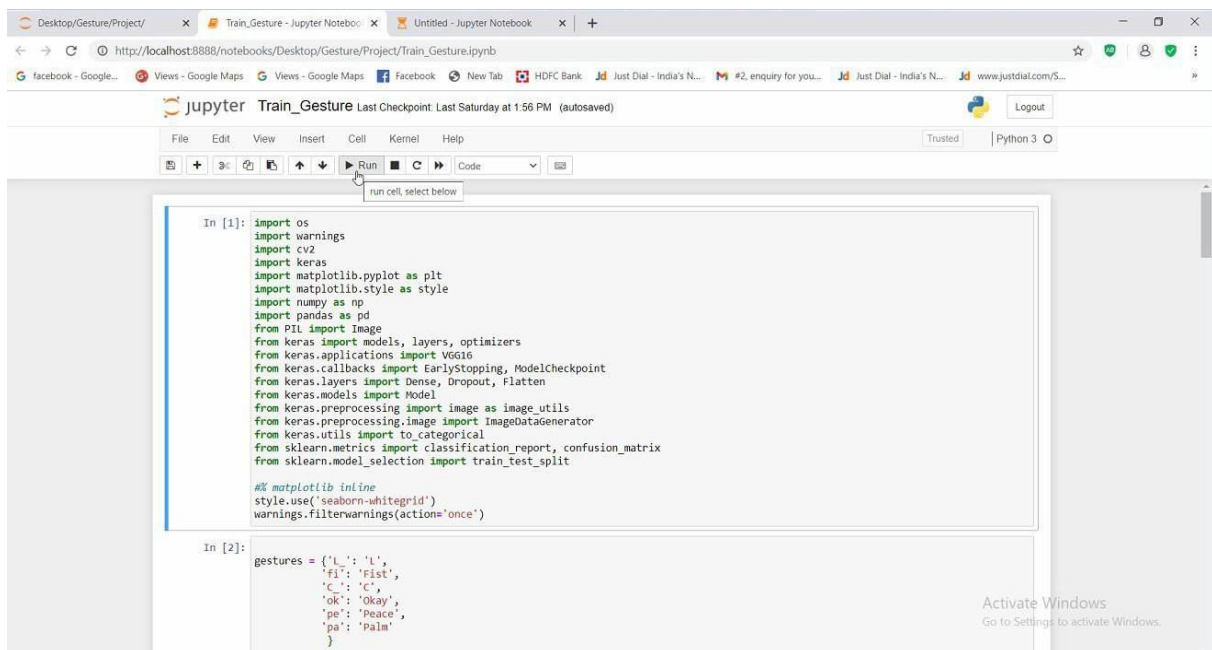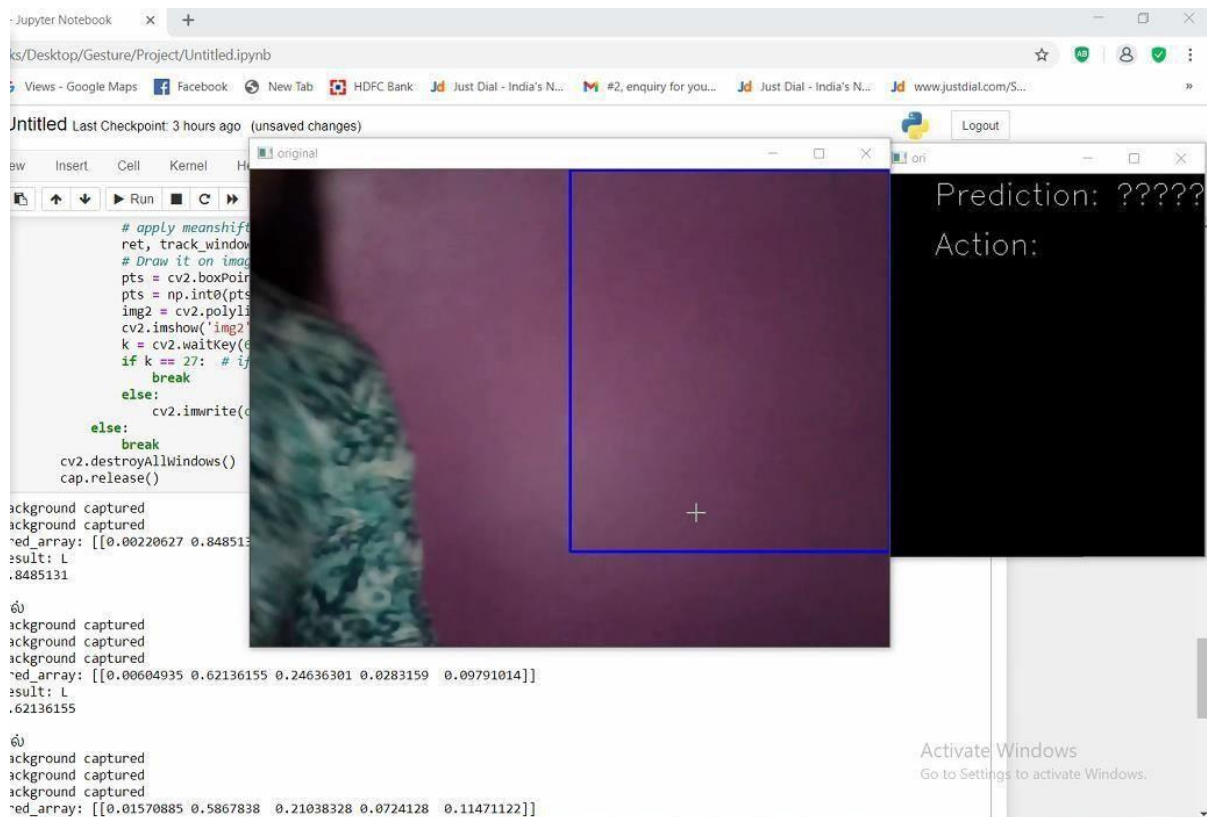
APPENDICES

APPENDICES

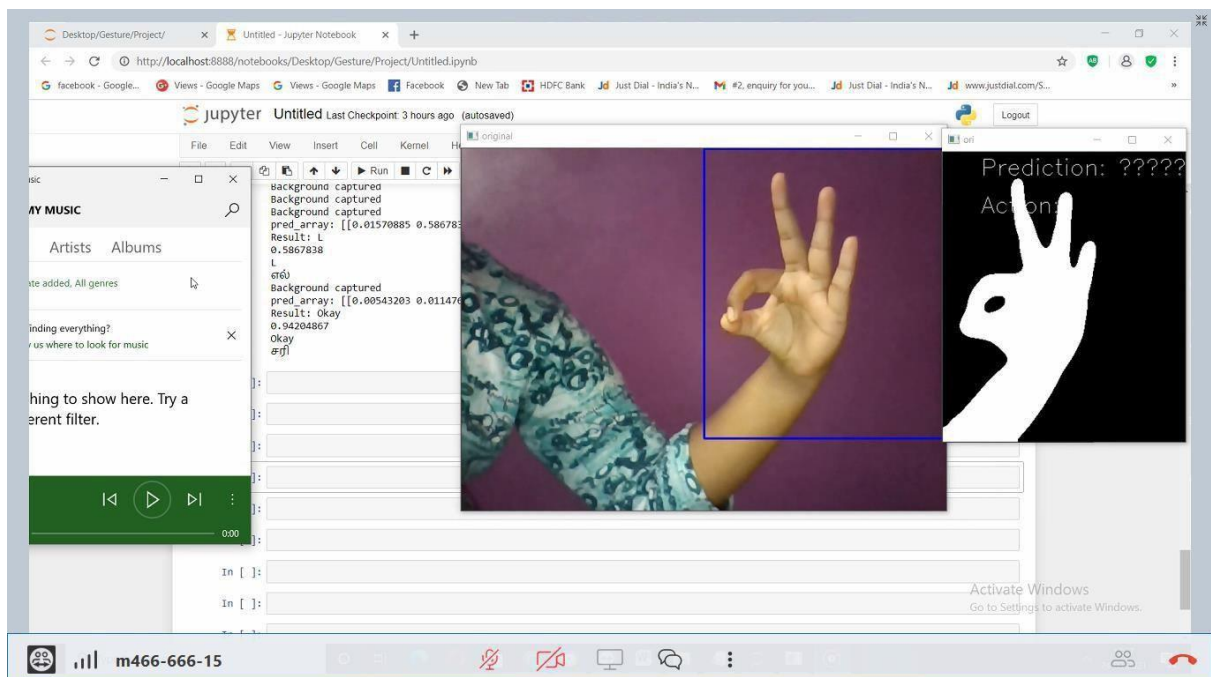## A.1 SAMPLESCREENS

Fig A.1.1. Jupyter Document Page

FigA.1.2. Training The System

FigA.1.3. Camera Capturing

FigA.1.4. Captured Gesture Converted To Voice And Text Message

Jupyter Untitled Last Checkpoint: 3 hours ago (autosaved)   Logout

File   Edit   View   Insert   Cell   Kernel   Help   Trusted   Python 3 ●

```python
            img2 = cv2.polylines(frame, [pts], True, (0, 255, 0), 2)
            cv2.imshow('img2', img2)
            k = cv2.waitKey(60) & 0xff
            if k == 27:   # if ESC key
                break
            else:
                cv2.imwrite(chr(k) + ".jpg", img2)
        else:
            break
    cv2.destroyAllWindows()
    cap.release()
```

```
Background captured
Background captured
pred_array: [[0.00220627 0.8485131  0.0520974  0.02773517 0.0694479 ]]
Result: L
0.8485131
L
எல்
Background captured
Background captured
Background captured
pred_array: [[0.00604935 0.62136155 0.24636301 0.0283159  0.09791014]]
Result: L
0.62136155
L
எல்
Background captured
Background captured
Background captured
pred_array: [[0.01570885 0.5867838  0.21038328 0.0724128  0.11471122]]
Result: L
0.5867838
L
எல்
Background captured
```

FigA.1.5. Backend Output

PUBLICATIONS

A.2

JOURNAL NAME - JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH(JETIR)

PAPER TITLE - CONVERSION OF GESTURES TO VOICE AND TEXT MESSAGE IN REGIONAL LANGUAGE

PUBLISHED PAPER ID - JETIRES06028

REGISTRATION ID - 307456

PUBLISHED IN - Volume 8 | Issue 4 | Year April-2021

DOI (Digital Object Identifier) - http://doi.one/10.1729/Journal.26856

PAGE NO - 127-130

AREA - Science & Technology

ISSN NUMBER - 2349-5162

PUBLISHED PAPER URL - https://www.jetir.org/view?paper=JETIRES06028

PUBLISHED PAPER PDF - https://www.jetir.org/papers/JETIRES06028

Certificate of Publication

The Board of

Journal of Emerging Technologies and Innovative Research (ISSN : 2349-5162)

Is hereby awarding this certificate to

S.Menaka

In recognition of the publication of the paper entitled

CONVERSION OF GESTURES TO VOICE AND TEXT MESSAGE IN REGIONAL LANGUAGE

Published In JETIR (www.jetir.org) ISSN UGC Approved (Journal No: 63975) & 7.95 Impact Factor

Published in Volume 8 Issue 4 , April-2021 | Date of Publication: 2021-04-20

Parin P
EDITOR

EDITOR IN CHIEF

JETIRES06028   Research Paper Weblink http://www.jetir.org/view?paper=JETIRES06028   Registration ID : 307456

Certificate of Publication

The Board of

Journal of Emerging Technologies and Innovative Research (ISSN : 2349-5162)

Is hereby awarding this certificate to

Sheshadri Roshni Sruthi

In recognition of the publication of the paper entitled

CONVERSION OF GESTURES TO VOICE AND TEXT MESSAGE IN REGIONAL LANGUAGE

Published In JETIR (www.jetir.org) ISSN UGC Approved (Journal No: 63975) & 7.95 Impact Factor

Published in Volume 8 Issue 4 , April-2021 | Date of Publication: 2021-04-20

Parin P
EDITOR

EDITOR IN CHIEF

JETIRES06028   Research Paper Weblink http://www.jetir.org/view?paper=JETIRES06028   Registration ID : 307456

Certificate of Publication

The Board of

Journal of Emerging Technologies and Innovative Research (ISSN : 2349-5162)

Is hereby awarding this certificate to

P.Preethi

In recognition of the publication of the paper entitled

CONVERSION OF GESTURES TO VOICE AND TEXT MESSAGE IN REGIONAL LANGUAGE

Pazin P
EDITOR

EDITOR IN CHIEF

ISSN
2349-5162

CONVERSION OF GESTURES TO VOICE AND TEXT MESSAGE IN REGIONAL LANGUAGE

V. Sathya Preiya, Associate Professor,
S.Menaka, P.Preethi,Sheshadri Roshni Sruthi,
Department of Computer Science Engineering,
Panimalar Engineering College, Chennai, Tamilnadu, India.

ABSTRACT:

Speech and text is the main medium for human communication. A person needs vision to access the information in a text. However those who have poor vision can gather information from voice. This paper proposes a camera based assistive text reading to help visually impaired person in reading the text present on the captured image. The faces can also be detected when a person enter into the frame by the mode control. The proposed idea involves text extraction from scanned image using Tesseract Optical Character Recognition (OCR) and converting the text to speech by e-Speak tool, a process which makes visually impaired persons to read the text. This is a prototype for blind people to recognize the products in real world by extracting the text on image and converting it into speech. Computer vision is one of the emerging technologies that can be used to aid visually impaired people for navigation (both indoor and outdoor), accessing printed material, etc. This paper describes an approach to extract and recognize text from scene images effectively using computer vision technology and to convert recognized text into speech so that it can be incorporated with hardware to develop Electronic travel aid for visually impaired people in future.

KEYWORDS: hand gesture recognition, voice conversion, gesture to speech, speech to gesture conversion.

I .INTRODUCTION:

Recent developments in computer software andrelated hardware technology have provided a value added service to the users. In everyday life, physical gestures are a powerful means of communication. They can economically convey a rich set of facts and feelings. For example, waving one's hand from side to side can mean anything from a "happy goodbye" to "caution". Use of the full potential of physical gesture is also something that most human computer dialogues lack.The task of hand gesture recognition is one the important and elemental problem in computer vision. With recent advances in information technology and media, automated human interactions systems are build which involve hand processing task like hand detection, hand recognition and hand tracking.This prompted my interest so I planned to make a software system that could recognize human gestures through computer vision, which is a sub field of artificial intelligence. The purpose of my software throughcomputer vision was to program a computer to "understand" a scene or features in an image.A first step in any hand processing system is to detect and localize hand in an image. The hand detection task was however challenging because of variability in the pose, orientation, location and

scale. Also different lighting conditions add furthervariability.

LITERATURE SURVEY:

Hand Gesture Recognition and Voice Conversion for Deaf and Dumb:

Sign language plays a major role for dump people to communicate with normal people. It is very difficult for mute people to convey their message to normal people. Since normal people are not trained on hand sign language. In emergency time conveying their message is very difficult. So the solution for this problem is to convert the sign language into human hearing voice. There are two major techniques available to detect hand motion or gesture such as vision and non-vision technique and convert the detected information into voice through raspberry pi. In vision based technique camera will be used for gesture detection and non- vision based technique sensors are used. In this project non- vision based technique will be used. Most of the dumb people are deaf also. So the normal people's voice can be converted into their sign language. In an emergency situation the message will automatically send to their relation or friends.

Design and implementation of a sign-to-speech/text system for deaf and dumb people:

This paper presents an approach for designing and implementing a smart glove for deaf and dumb people. There have been several researches done in order to find an easier way for non-vocal people to communicate with vocal people and express themselves to the hearing world. Developments have been made in sign language but mainly in American Sign Language. This research aims to develop a sign to Arabic language translator based on smart glove interfaced wirelessly with microcontroller and text/voice presenting devices. An approach has been developed and programmed to display Arabic text. The whole system has been implemented, programmed, cased and tested with very good results.

Gesture Aided Speech for Deaf and Mute:

Speech impaired people make use of sign language to communicate along with normal people. Common people also face difficulties to understand the gesture language. In order to minimize these real time issues, an attempt has been made to develop a system which consists of Flex sensors attached with the Data gloves along all fingers. These will be wearable gloves converts hand gestures into an audio output to interpret the expression. This reduces the communication gap that exists between mute and ordinary people. The proposed system plays corresponding recorded voice on an android phone connected to the hardware system via Bluetooth Module as an interpretation of the gesture. It is

achieved by integrating flex sensor and 8051 microcontroller. The phone will use a text to speech converter to produce the output of corresponding gesture. This system offers high reliability and fast response as microcontroller works at a speed of 12 MHz. Processing time of microcontroller is much less than the time taken for a human to change from one gesture to another, the system works efficiently in real time scenario hence making the system faster.

Implementation of gesture based voice and language translator for dumb people:

Dumb persons communicate through gestures which are not understood by the majority of people. Gesture is a movement of part of the body, especially a hand or the head, to express an idea or meaning. This paper proposes a system that converts gestures given by the user in the form of English alphabets into corresponding voice and translates this English voice output into any other Microsoft supported languages. The system consists of MPU6050 for sensing gesture movement, Raspberry pi for processing, three button Keypad and speaker. It is implemented by using trajectory recognition algorithm for recognizing alphabets. Raspberry pi generates voice output for the text in multiple languages using voice RSS and Microsoft translator. When tested, the system recognized A-Z alphabets and generated voice output based on the gestures in multiple languages.

EXISTING SYSTEM:

In recent decades, due to computer software and hardware technologies of continuous innovation and breakthrough, the social life and information technology have a very close relationship in the twenty-first century. In the future, especially the interfaces of consumer electronics products (e.g. smart phones, games and infota1inment systems) will have more and more functions and be complex. How to develop a convenienthuman-machineInterface (HumanMachine Interaction/Interface, HMI) for each consumer electronics product has become an important issue. The traditional electronic input devices, such as mouse, keyboard, and joystick are still the most common interaction way. However, it does not mean that these devices are the most convenient and natural input devices for most users. Since ancient times, gestures are a major way for communication and interaction between people. People can easily express the idea by gestures before the invention of language. Nowadays, gestures still are naturally used by many people and especially are the most major and nature interaction way for deaf people [1]. In recent years, the gesture control technique has become a new developmental trend for many humanbased electronics products, such as computers, televisions, and games. This technique let people can control these classifiers selection are a major issue in most researches. The third stage is to analyze sequential gestures to identify users' instructs or behaviors.

PROPOSED SYSTEM:

Most gesture recognition methods usually contain three major stages. The first stage is the object detection. The target of this stage is to detect hand objects in the digital images or videos. Many environment and image problems are needed to solve at this stage to ensure that the hand contours or regions can be extracted precisely to enhance the recognition accuracy. Common image problems contain unstable brightness, noise, poor resolution and contrast. The better environment and camera devices can effectively improve these problems. However, it is hard to control when the gesture recognition system is working in the real environment or is become

product. Hence, the image processing method is a better solution to solve these image problems to construct an adaptive and robust gesture recognition system. The second stage is object recognition. The detected hand objects are recognized to identify the gestures. At this stage, differentiated features and effective.
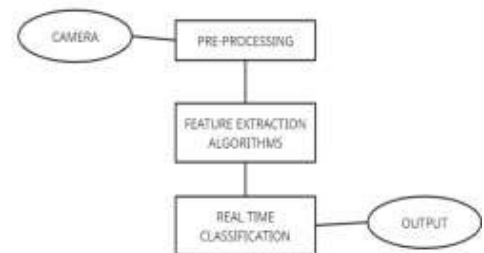
BLOCK DIAGRAM:
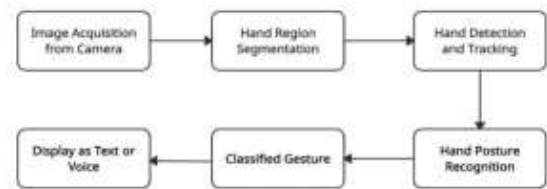
SYSTEM IMPLEMENTATION:

Hand gesture recognition system can be dividedinto following modules :

- Pre-processing
- Featureextraction of the processedimage
- Real timeclassification

Value measure intensity or brightness. This is well enough to choose single colour but it ignores complexity of colour appearance. It trade off computation speed mean computationally expensive and perceptual relevance.

a) Pre Processing

Like many other pattern recognition tasks, pre-processing is necessary for enhancing robustness and recognition accuracy.

The pre-processing prepares the image sequence for the recognition, so before calculating the diagonal Sum and other algorithms, a pre-processing step is performed to get the appropriate image, which is required for real time classification. So it consists of some steps. The net effect of this processing is to extract the hand only from the given input because once the hand is detected from the given input it can be recognized easily. So pre- processing step mainly consists of following tasks:

- SkinModeling.
- Removal of Background.
- Conversion from RGB tobinary.
- HandDetectionmodules.

SkinModelling:

There are numerous method used for skin detection such as RGB (Red, Green, Blue), YCbCr (Luminance Chrominance) and HSV (Hue, Saturation, Value).

69a

- **RGB:**

RGB is a 3D color space pixel where each pixel has combination of three colors Red, Green and Blue at specific location. This technique widely used in image processing for identifying skin region.

YCbCr (Luminance Chrominance):

This color space is used in digital video color information represent two color Cb and Cr. Cb is difference between Blue and Cr is difference between Red component references of value. This is basically RGB transformation to YCbCr for separation of luminance and chrominance for color modelling.

HSV (Hue, Saturation and Value):

In HSV, Hue detect dominant color and Saturation define colourfulness whilst

My approach for this thesis is to work with RGB to binarization techniques to Explicitly Defined skin Region.

Skin Detection:

The skin color detection is one of important goal in hand gesture recognition. Skin color detection decision rules which we have to build that will discriminate between skin portion and non-skin portion pixels. This is accomplished usually by metric introduction, which measure distance of the pixel color. This metric type is knows as skin modelling.

Explicitly Defined Skin Region:

Following are some common ethnic skin groups and there RGB color space:

Different Ethnic Group Skin Patches

To build a skin classifier is to define explicitly through a number of rules the boundaries of skin color cluster in some color space. The advantage of this method is the simplicity of skin detection rules that leads to the construction of very rapid classifier.

In this classifier threshold defined to maximize the chance for recognizing the skin region for each color. If we see in that Red color in every skin sample is greater than 95, Green is greater than 40 and Blue is greater than 20 in. So threshold can make this classifier easily detect almost all kind of skin.

This is one of the easiest methods as it explicitly defines skin-color boundaries in different color spaces. Different ranges of thresholds are defined according to each color space components in as the image pixels that fall between the predefined ranges are considered as skin pixels. The advantage of this method is obviously the simplicity which normally avoids of attempting too complex rules to prevent over fitting data. However, it is important to select good color space and suitable decision rules to achieve high recognition rate with this method.

Removal of Background:

I have found that background greatly affects the results of hand detection that's why I have decided to remove it. For this I have written our own code in spite of using any built-in ones.

BeforeAfter

Removal of Background

a) Conversion from RGB to Binary:

All algorithms accept an input in RGB form and then convert it into binary format in order to provide ease in recognizing any gesture and also retaining the luminance factor in an image.

b) Hand detection:

Image could have more than one skin area but we required only hand for further process. For this I choose criteria image labeling which is following:

c) Labeling:

To define how many skin regions that we have in image is by labelling all skin regions. Label is basically an integer value have 8 connecting objects in order to label all skin area pixel. If object had label then mark current pixel with label if not then use new label with new integer value. After counting all labelled region (segmented image) I sort all them into ascending order with maximum value and choose the area have maximum value which I interested because I assume that hand region in bigger part of image. To separate that region which looked for, create new image that have one in positions where the label occurs and others set to zero.

BeforeAfter

Labeling Skin Region

MODULES:
- Dataset collection.
- Dataset pre-processing.
- Machine Learning Training.
- Capture Real time video.
- Eliminate Background.
- Identify Gesture.

Dataset Collection:
Dataset collection involves collecting or creating test dataset for gesture and respective terminology. This is very exhaustive task as we need to collect more samples for gesture. We collect dataset from kaggle website for the project in scope.

Dataset Pre-Processing:
Dataset preprocessing involves removing noise in images and converting to numpy arrays.

Machine Learning:
Machine learning involves training and classification using

70

CNN algorithms.

**Capture Real Time Video & Gesture Detection:**
We use opencv to capture and process video. Here we capture video and pre-process frames and fit in model to get the gesture

Background Elimination:
We will find the contours to detect edges of the main object and create a mask with numpy zeros for the background and then combine the mask and the image using bitwise and operator.

RESULT:
In result, after execution of the codecamera will be capturing the gesture and it will eliminate backgroundthen output will be shared in voice and text.
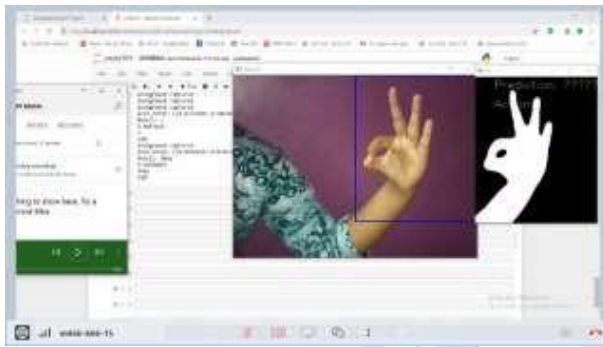
CONCLUSION:
The aim of this paper is to build a system to achieve real time gesture recognition with Indian Sign Language alphabets as the use case. In order to do so I created multiple versions of datasets, trained corresponding models with high accuracy and different methods based on camera and background settings. The project has resulted in a pipeline which can be used to train any gesture recognition application as only the dataset has to be changed and other steps remain the same.

REFERENCES:
1. J.Jenkinwinston (96207106036), M.Maria Gnanam (96207106056), R.Ramasamy (96207106306), Anna University ofTechnology, Tirunelveli : Hand Gesture Recognitionsystem Using HaarWavelet.
2. Laura Dipietro, Angelo M. Sabatini, *Senior Member, IEEE*, and Paolo Dario, *Fellow, IEEE,* A Survey of Glove-Based Systems and TheirApplications.
3. Kay M. Stanney HANDBOOK OF VIRTUAL ENVIRONMENTS Design, Implementation, and Applications, Gesture Recognition Chapter #10 byMatthew Turk
4. Daniel Thalman, Gesture Recognition Motion Capture, Motion Retargeting, and ActionRecognition
5. Hatice Gunes, Massimo Piccardi, Tony Ja, 2007, Research School of Information Sciences and Engineering Australian National University Face and Body Gesture Recognition for a Vision-Based Multimodal Analyzer.
6. J. Heinzm ann and A. Zelinsky Robust Real - Time Face Tracking and Gesture Recognition
7. Vladimir Vezhnevets, Vassili Sazonov, Alla Andreeva, Moscow State University A Survey on Pixel-Based Skin Color Detection Techniques.
8. TEY YI CHI, Universiti Teknologi Malaysi, FUZZY SKINDETECTION.
9. Robyn Ball and Philippe Tissot, Texas A&M University, Demonstration of Artificial Neural Network in Matlab.
10. Howard Demuth, Mark Beale, Neural NetworkToolbox.
11. Ari Y. Benbasat and Joseph A. Paradiso in MIT Media Laboratory, Cambridge An Inertial Measurement Framework for Gesture Recognition andApplications.
12. Peggy Gerardin, Color Imaging Course Winter Semester 2002-2003, Color Image Segmentation.
13. Michael Nielsen, Moritz Störring, Thomas B. Moeslund, and Erik Granum, March 2003, A procedure for developing intuitive and ergonomic gesture interfaces for man- machineinteraction.
14. Oğuz ÖZÜN1, Ö. Faruk ÖZER2, C. Öncel TÜZEL1, Volkan ATALAY, A. Enis ÇETİN2, Dept. of Computer Engineering, Middle East Technical University, Ankara, Turkey, Dept. of Electrical Engineering, Bilkent University, Ankara,Turkey
Faculty of Engineering, Sabanci University, Istanbul, Turkey, Vision based single stroke character recognition for wearable computing.
15. Melinda M. Cerney1, Judy M. Vance, March 28, 2005, Gesture Recognition in Virtual Environments: A Review and Framework for FutureDevelopment.
16. Ashutosh Saxena, Aditya Awasthi, Vaibhav Vaish, and SANKET: Interprets your HandGestures.
17. Yoichi Sato, Institute of Industrial Science, the University of Tokyo, Makiko Saito Hideki Koike, Graduate School of Information Systems, University of Electro- Communications, Tokyo Real-Time Input of 3D Pose and Gestures of a User's Hand and Its Applications forHCI.
18. Mohammad Al-aqrabawi Fangfang Du, 11th March" 2000, Human Skin Detection Using ColorSegmentation.
19. Manuel Cabido Lopes Jos´e Santos-Victor Instituto de Sistemas e Rob´otica, Institution Superior T´ecnico, Lisbon, Portugal in IROS Workshop on Robot Programming by demonstration, Las Vegas, SA, Oct 31st, 2003 Motor Representations for Hand Gesture Recognition andImitation.
20. Raymond Lockton and Andrew W. Fitzgibbon, Department of Engineering Science University of Oxford. Real-time gesture recognition using deterministicboosting.

71

REFERENCES

REFERENCES

[1]. J.Jenkin winston, M.MariaGnanam, R.Ramasamy, "Hand Gesture Recognition system Using HaarWavelet", Anna University of Technology, Tirunelveli.

[2]. Laura Dipietro, Angelo M. Sabatini, Senior Member, IEEE, and Paolo Dario, Fellow, "A Survey of Glove-Based Systems and Their Applications", IEEE.

1. Kay M. Stanney HANDBOOK OF VIRTUAL ENVIRONMENTS Design, Implementation, and Applications, Gesture Recognition Chapter #10 byMatthew Turk

2. Daniel Thalman, Gesture Recognition Motion Capture, Motion Retargeting, and Action Recognition.

3. HaticeGunes, Massimo Piccardi, Tony Ja, 2007, Research School of Information Sciences and Engineering Australian National University Face and Body Gesture Recognition for a Vision-Based Multimodal Analyzer.

4. J. Heinzmann and A. Zelinsky Robust Real - Time Face Tracking and Gesture Recognition.

5. Vladimir Vezhnevets, VassiliSazonov, AllaAndreeva, Moscow State University A Survey on Pixel-Based Skin Color Detection Techniques.

6. TEY YI CHI, UniversitiTeknologiMalaysi, FUZZY SKINDETECTION.

7. Robyn Ball and Philippe Tissot, Texas A&M University, Demonstration of Artificial Neural Network in Matlab.

8. Howard Demuth, Mark Beale, Neural Network Toolbox.

9. Ari Y. Benbasat and Joseph A. Paradiso in MIT Media Laboratory,Cambridge An Inertial Measurement Framework for Gesture Recognition and Applications.

10. Peggy Gerardin, Color Imaging Course Winter Semester 2002-2003, Color Image Segmentation.

11. Michael Nielsen, Moritz Störring, Thomas B. Moeslund, and Erik Granum, March 2003, A procedure for developing intuitive and ergonomic gesture interfaces for man- machine interaction.

12. Melinda M. Cerney1, Judy M. Vance, March 28, 2005, Gesture Recognition in Virtual Environments: A Review and Framework for Future Development.

13. Ashutosh Saxena, Aditya Awasthi, Vaibhav Vaish, and SANKET: Interprets the Hand Gestures.

14. Yoichi Sato, Institute of Industrial Science, the University of Tokyo, Makiko Saito Hideki Koike, Graduate School of Information Systems, University of Electro-Communications, Tokyo Real-Time Input of 3D Pose and Gestures of a User's Hand and Its Applications for HCI.

15. Mohammad Al-aqrabawi Fangfang Du, 11 th March" 2000, Human Skin Detection Using Color Segmentation.

16. Manuel Cabido Lopes Jos´e Santos-Victor Instituto de Sistemas e Rob´otica, Institution Superior T´ecnico, Lisbon, Portugal in IROS Workshop on Robot Programming by demonstration, Las Vegas, SA, Oct 31 st , 2003 Motor Representations for Hand Gesture Recognition and Imitation.