A project report on

HAND DETECTION AND TRACKING USING OPENCV PYTHON

Submitted in Partial fulfillment of the Requirements for the Award of the Degree of

BACHELOR OF TECHNOLOGY

in

ELECTRONICS & COMMUNICATION ENGINEERING

by

NITYA SUMANVIKA GANGIPAMULA	**17A91A04D1**
SURYA DEEPTHI KUMPATLA	**17A91A04E4**
MANASA SEEPANA	**17A91A04G5**
VEERA AVINASH GUDHE	**17A91A04D5**
JOY KISHORE GADIPE	**17A91A04C9**

Under the Esteemed guidance of

Mr. S. BALA GANGADHAR TILAK BABU, M.Tech., (Ph.D.)
Associate Professor

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

ADITYA ENGINEERING COLLEGE (A)

(Approved by AICTE, Permanently Affiliated to JNTUK & Accredited by NBA, NAAC with 'A' Grade. Recognized by UGC under the sections 2(f) and 12(B) of the UGC act 1956)

Aditya Nagar, ADB Road, Surampalem – 533 437

(2017-2021)

ADITYA ENGINEERING COLLEGE (A)

(Approved by AICTE, Permanently Affiliated to JNTUK & Accredited by NBA, NAAC with 'A' Grade. Recognized by UGC under the sections 2(f) and 12(B) of the UGC act 1956)

Aditya Nagar, ADB Road, Surampalem – 533 437

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

Certificate

This is to certify that the project report entitled **"HAND DETECTION AND TRACKING USING OPENCV PYTHON"**

being submitted by

NITYA SUMANVIKA GANGIPAMULA	**17A91A04D1**
SURYA DEEPTHI KUMPATLA	**17A91A04E4**
MANASA SEEPANA	**17A91A04G5**
VEERA AVINASH GUDHE	**17A91A04D5**
JOY KISHORE GADIPE	**17A91A04C9**

for the partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in Department of Electronics & Communication Engineering of Aditya Engineering College to Jawaharlal Nehru Technological University, Kakinada is a record of bonafide work carried out by them under the guidance and supervision during Academic Year of 2020-21.

Project Guide | **Head of the Department**

Mr. S. Bala Gangadhar Tilak Babu | **Mr. Satyanarayana**

External

ACKNOWLEDGEMENT

We take this opportunity as a privilege to thank all individuals without whose support and guidance we could not have completed our project in this stipulated period of time.

We express our deep sense of gratitude to our guide **Mr. S. Bala Gangadhar Tilak Babu** for his valued suggestions and inputs during the course of the project work, readiness for consultation at all times, his educative comments and inputs, his concern and assistance even with practical things have been extremely helpful.

We highly indebted to our Head of the Department **Mr. V. Satyanarayana** for his motivational guidance and the vision in providing the necessary resources and timely inputs.

We are also thankful to **Dr. M. Sreenivasa Reddy,** Principal, Aditya Engineering College for providing appropriate environment required for this project and thankful to Faculty of Electronics and Communication Engineering Department for the encouragement and cooperation for this successful completion of the project.

(G. Nitya Sumanvika 17A91A04D1)

(K. Surya Deepthi 17A91A04E4)

(S. Manasa 17A91A04G5)

(G. Veera Avinash 17A91A04D5)

(G. Joy Kishore 17A91A04C9)

CONTENTS

NOMENCLATURE

CV	Camera Vision
HCI	Human Computer Interaction
RGB	Red Green Blue
HSV	Hue Saturation Value
BLOB	Binary Large Object
TOF	Time of Flight
IOR	Importer of Record
MRI	Magnetic Resonance Imaging
CT	Computed Tomography
ANN	Artificial Neural Network
DSP	Digital Signal Processing
MATLAB	Matrix Laboratory

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Man, always wanted to create wonders using technology. Gradually technology became a part of man's day-to-day life, not only for the proper utilization for the man kind, but also for relaxation, fun and entertainment. As per the increasing technology many projects can be made uniquely and differently that are meant both for effective utilization and entertainment for the human world. Man, always strived to do things differently using artificial technology which is the trending technology in the present-day scenario. He started using that for various applications in our day today life. From one among such wonderful ideas there born this "HAND GESTURE RECOGNITION AND FINGER COUNTING". In this we will see how a human hand will be detected when it is placed in front of the camera and how the gestures are recognized.

Hand gesture recognition is very significant for human-computer interaction. In this project, we present a novel real-time method for hand gesture recognition. In this framework, the hand region which is detected in the selected region, is extracted from the background with the background subtraction method. Then the image is to be converted from BGR to HSV format and masking is done. Then, the palm and fingers are segmented so as to detect and recognize the fingers. Finally, a rule-based classification is done to detect and display the number of fingers projected in the detected hand.

Moreover, it is not only meant for fun and entertainment but can also be used in our daily life for various applications. In this project we will deeply know about the requirements, working, advantages, applications and the logic present behind its detection.

Chapter-1

INTRODUCTION

1.1 INTRODUCTION:

As we know, the vision-based technology of hand gesture recognition is an important part of human-computer interaction (HCI). Over the past few decades, we have been communicating to the computer using keyboard and mouse. In today's generation we are using voice to text (Speech recognition) methods to interact with computers. But we all know that gestures play an important role in communicating whether it may be a person or a machine. Gesture is a symbol of expression of a person's condition or his opinion like all the best, good luck, okay…...etc., Every gesture has its unique way of expression. So, gestures can be used as a tool for human computer interaction. That's how the concept of communicating to the computer using handmade gestures was born. But to achieve this wonderful idea a few requirements are to be fulfilled. Like the necessary software and the required environment for the working of the project. This can be achieved only by Artificial Intelligence. So, then Artificial Technology came into utilization.

So, for the successful working of the project we've got to use artificial technology. For that we require coding platforms like python and in that we require some of the packages to provide the environment required like Open cv for image recognition, NumPy for n dimensional array objects and math for the performing of operations like square roots and other required mathematical operations.

By using the above requirements and necessary platforms and packages Human-Computer Interaction can be done using hand gestures. The proposed model is capable of detecting the hand gesture and counting the fingers displayed infront of the camera successfully.

1.2 MOTIVATION:

The main motivation behind this project is to make a system to recognize hand gestures using programming language python and OpenCV. As er know that human computer interaction plays a vital role in interacting with people. We are well aware of the fact that apart typing, speaking, gestures make an important role in communicating. Also, there are many sign languages to interact with people apart from communicating through various languages. So, why not use them for interacting with the technology?

Gestures especially hand gestures play an important role in our day-to-day life. So, if we use it in our HCI (Human Computer Interaction) we can create wonders using that technology. Thus, this wonderful idea of hand gesture recognition and interaction to technology through it took birth. People who cannot communicate orally may use this to communicate with the technology. And moreover, we can make technology do us things by using this HCI.

Finger counting is considered as the simple and easiest task that can be done using hands. So, if this was achieved first using technology, further it can be developed and improves to create more and more wonders. And those finger gestures can be used to indicate some sign that we want the system to do. Thus, the idea of hand detecting and tacking took place using simple Python language and OpenCV.

1.3 OBJECTIVE OF THE PROJECT:

The main objective is to program a human hand detecting, gesture recognizing and finger counting code using programming language python with the OpenCV library. The code must be compatible to detect and display the hand in the masked format and show the convex hull image of it. The code has to count the number of fingers displayed infront of the camera and it has to display the count in the python console.

The program must be able to capture the video, convert it into images, reduce the background noise, detect the hand, count the fingers and display the outputs. All this has to happen continuously till the user stops it externally.

1.4 SOFTWARE REQUIREMENTS:

Human Computer Interaction through camera is impossible without OpenCV (Camera Vision) use. So, to provide the required environment for the OpenCV one of the today's trending languages Python is used.

Python is the trending yet simple language which is compatible to ant operating system. The libraries included in this are:

- Open CV
- NumPy
- Math
- Copy
- Imutils

1.5. ORGANISATION OF THESIS:

The work carried out has been summarized in six Chapters.

In Chapter 1: An overview of the Project is discussed.

In Chapter 2: History of hand detection is discussed along with the Literature survey.

In Chapter 3: The Required components are clearly discussed.

In Chapter 4: Proposed Methodology is discussed including code and background processes with the help if a flowchart.

In Chapter 5: Results of the proposed method and applications are discussed

In Chapter 6: Conclusion is made based on the project and future directions are discussed.

Chapter-2

LITERATURE REVIEW ON HAND DETECTION TECHNIQUES

2.1 HISTORY OF HAND DETECTION:

1) Hand detection is a crucial preprocessing procedure for many human hand related computer vision tasks, such as hand gesture recognition.

2) The hand gesture is invented in early 1980s. the first step was to use voice control and special hand gloves to interact with objects.

3) In 1960s researchers started using tablets and special pencils that captured writing, painting devices.

4) Later, in 1969 engineer 'MYRON KRUEGER' started working with virtual reality prototypes. this approach would later be referred as natural interaction.

5) In 1990s work on identifying gestures in images and video with computer vision methods are increased.

6) In 1983 the First glove that actually recognized hand positions. This data glove consisted of a light weight glove with optical fibers attached to the back of the fingers.

7) Later in 1989 and 1995, the 'power glove' and 'super glove' appeared and they both used materials of variable electric resistance in order to measure the fingure flexions.

8) These data gloves, super gloves used different sensor types to capture hand motion and position by detecting the location of palm and fingers.

9) These provided good outcomes. but slowly that make them unsuitable for the elderly who experience discomfort and confusion due to wire connection problems.

10) In addition, it also causes skin damage, infection in people with sensitive skin or those suffering burns.

11) Some of these problems were addressed in a study by 'Lamberti and Camastra' who develop a computer vision system based on colored marked gloves.

12) These drawbacks led to the development of promising and cost-effective techniques that did not require gloves to be worn. These techniques are called camera vision-based sensor technologies.it is easier than ever to detect hand gestures.

13) It can be used under a wide range of applications like 'clinical operations, sign language, robot control, virtual environments, home automation'.

14) The primary goal in studying gesture recognition is to introduce a system that can detect specific human gestures and use them to convey information and control purposes.

15) Two approaches are generally used to interpret gestures for HCI applications

- *Hand Gestures Based on Instrumental Glove Approach:*

  The wearable glove-based sensors can be used to capture hand motion and position. In addition, they can easily provide the exact coordinates of palm and finger locations, orientation and configurations by using sensors attached to the gloves.

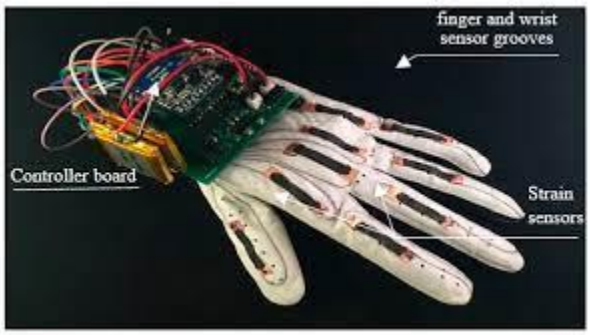

Figure 2.1.1: Glove based sensors

- *Hand Gestures Based on Computer Vision Approach:*

The camera vision-based sensor is a common and applicable technique because it provides contactless communication between humans and computers. Different configurations of cameras can be utilized, such as monocular, TOF and IOR.

16) Using computer vision techniques to identify gestures. where the user performs specific gesture by single or both hand infront of camera which connect with system that involve different possible techniques. They are

- *Color Based Recognition Using Glove Marker:*

This method uses a camera to track the movement of hand using a glove with different glove marks. this method has been used for interaction with 3D models.

- *Color Based Recognition of Skin Color:*

Skin color detection is one of the most popular for hand segmentation and is used in wide range of applications, such as object classification, person movement tracking, HCI applications.

Figure 2.1.2: Skin color-based hand detection

- *Appearance Based Recognition:*

This method depends on extracting the image features in order to model visual appearance such as hand and comparing these parameters

with feature extracting from the input image features.

- *Motion Based Recognition:*

Motion based recognition can be utilized for detection purposes, it can be extracting the object through a series of image frames.

- *Skeleton Based Recognition:*

The skeleton-based recognition specifies model parameters which can improve the detection of complex features. The most common feature used is joint orientation, the space between the joints.

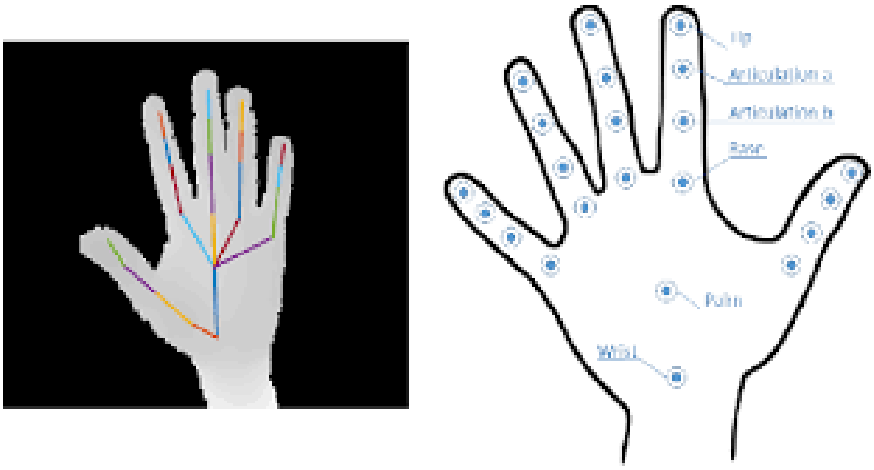

Figure 2.1.3: Skeleton based hand detection

- *Depth Based Recognition:*

A depth camera provides 3D geometric information about the object. the 3D data from a depth camera directly reflects the depth field if compared with a colored image which contains only a projection.

- *3d Model Based Recognition:*

The 3D model essentially depends on 3D kinematic hand model which has a large degree of freedom, where hand parameter estimation obtained by comparing the input image with the two-

dimensional appearance projected by 3-dimensional hand model.

- *Deep Learning Based Recognition:*

The artificial intelligence offers a good and reliable technique used in a wide range of modern applications because of using a learning role principle. the deep learning used multi layers for learning data and gives a good prediction out result.

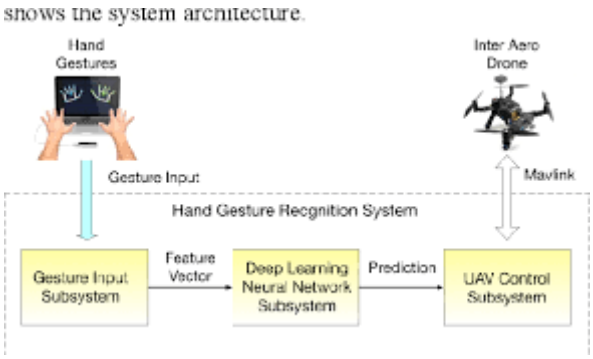

Figure 2.1.4: Deep-learning based hand detection

17) Applications of hand gesture recognition system

- *Clinical And Health:*

This is achieved by using a medical imaging system such as MRI, CT OR X-RAY SYSTEM.

- *Sign Language Recognition:*

Sign language is an alternative method used by people who are unable to communicate others by speech.

- *Robot Control:*

Robot technology is used in many applications fields such as industry, assistive services, sports and entertainment.

- *Virtual Environment:*

  Virtual environments are based on a 3D model that needs a 3D gesture recognition system in order to interact in real time as a HCI.

- *Home Automation:*

  Hand gestures can be used efficiently for home automation. Shaking a hand or performing some gestures can easily enable control of fans, radio etc.,

2.2 INTRODUCTION TO HAND DETECTION:

Hand gestures are an aspect of body language that can be conveyed through the center of the palm, the finger position and the shape constructed by the hand. Hand gestures can be classified into static and dynamic. As its name implies, the static gesture refers to the stable shape of the hand, whereas the dynamic gesture comprises a series of hand movements such as waving. There are a variety of hand movements within a gesture; for example, a handshake varies from one person to another and changes according to time and place. The main difference between posture and gesture is that posture focuses more on the shape of the hand whereas gesture focuses on the hand movement. The main approaches to hand gesture research can be classified into the wearable glove-based sensor approach and the camera vision-based sensor approach. Hand gestures offer an inspiring field of research because they can facilitate communication and provide a natural means of interaction that can be used across a variety of applications. Previously, hand gesture recognition was achieved with wearable sensors attached directly to the hand with gloves. These sensors detected a physical response according to hand movements or finger bending. The data collected were then processed using a computer connected to the glove with wire. This system of

glove-based sensor could be made portable by using a sensor attached to a microcontroller. As illustrated in Figure 1, hand gestures for human–computer interaction (HCI) started with the invention of the data glove sensor. It offered simple commands for a computer interface. The gloves used different sensor types to capture hand motion and position by detecting the correct coordinates of the location of the palm and fingers. Various sensors using the same technique based on the angle of bending were the curvature sensor, angular displacement sensor, optical fiber transducer, flex sensors and accelerometer sensor. These sensors exploit different physical principles according to their type.

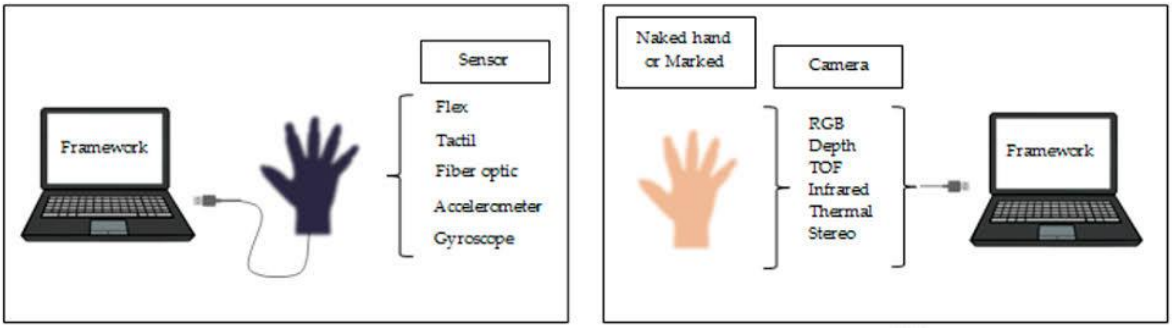

Figure 2.2.1: a)Glove based attached to sensor either connected to computer or portable b)computer vision-based camera using a marked glove or just naked hand

Although the techniques mentioned above have provided good outcomes, they have various limitations that make them unsuitable for the elderly, who may experience discomfort and confusion due to wire connection problems. In addition, elderly people suffering from chronic disease conditions that result in loss of muscle function may be unable to wear and take off gloves, causing them discomfort and constraining them if used for long periods. These sensors may also cause skin damage, infection or adverse reactions in people with sensitive skin or those suffering burns. Moreover, some sensors are quite expensive. Some of these problems were addressed in a study by Lamberti and Camastra, who developed a computer vision system based on colored marked

gloves. Although this study did not require the attachment of sensors, it still required colored gloves to be worn. These drawbacks led to the development of promising and cost-effective techniques that did not require cumbersome gloves to be worn. These techniques are called camera vision-based sensor technologies. With the evolution of open-source software libraries, it is easier than ever to detect hand gestures that can be used under a wide range of applications like clinical operations, sign language, robot control, virtual environments, home automation, personal computer and tablet, gaming. These techniques essentially involve replacement of the instrumented glove with a camera. Different types of cameras are used for this purpose, such as RGB camera, time of flight (TOF) camera, thermal cameras or night vision cameras. Algorithms have been developed based on computer vision methods to detect hands using these different types of cameras.

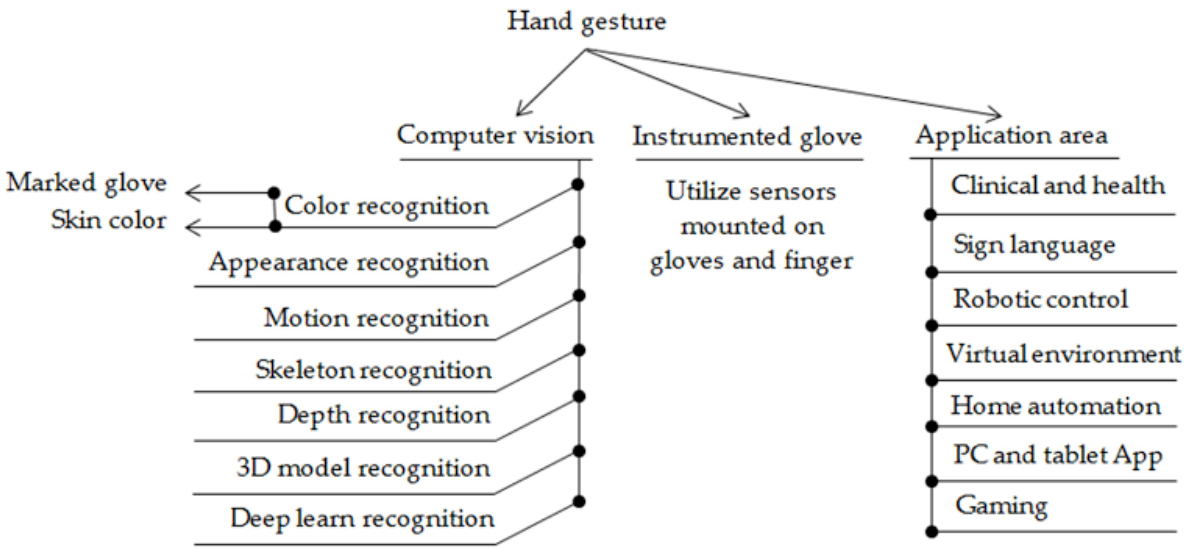

Figure 2.2.2: Classification methods conducted

Hand Gesture Methods

The primary goal in studying gesture recognition is to introduce a system that can detect specific human gestures and use them to convey information or for command-and-control purposes. Therefore, it includes not only tracking of human movement, but also the interpretation of that movement as significant commands. Two approaches are generally used to interpret gestures for HCI applications. The first approach is based on data gloves (wearable or direct contact) and the second approach is based on computer vision without the need to wear any sensors.

- *Hand Gestures Based on Instrumented Glove Approach*

The wearable glove-based sensors can be used to capture hand motion and position. In addition, they can easily provide the exact coordinates of palm and finger locations, orientation and configurations by using sensors attached to the gloves. However, this approach requires the user to be connected to the computer physically, which blocks the ease of interaction between user and computer. In addition, the price of these devices is quite high. However, the modern glove-based approach uses the technology of touch, which more promising technology and it is considered Industrial-grade haptic technology. Where the glove gives haptic feedback that makes user sense the shape, texture, movement and weight of a virtual object by using microfluidic technology.

- *Hand Gestures Based on Computer Vision Approach*

The camera vision-based sensor is a common, suitable and applicable technique because it provides contactless communication between humans and computers. Different configurations of cameras can be utilized, such as monocular, fisheye, TOF and IR. However, this technique involves several challenges, including lighting variation, background issues, the effect of occlusions, complex background, processing time traded against resolution and frame rate and foreground or background objects presenting the same skin color tone or otherwise appearing as hands. These challenges will be discussed in the following sections. A simple diagram of the camera vision-based sensor for extracting and identifying hand gestures is presented in Figure

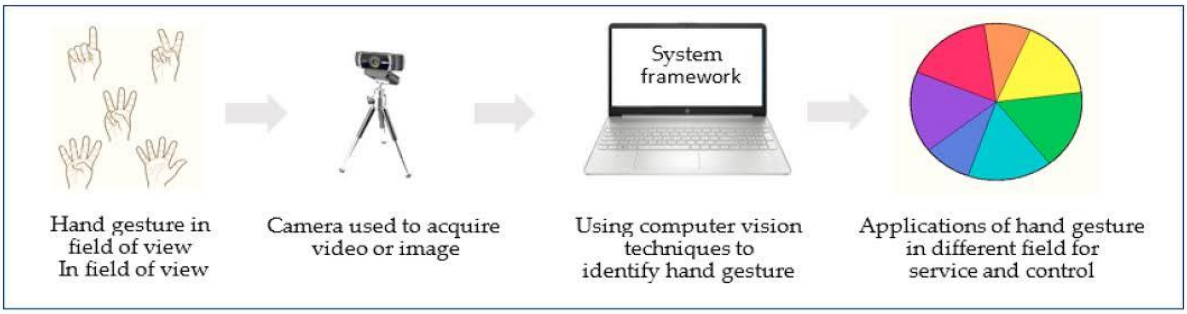

Figure 2.2.3: Computer vision techniques to identify gestures

In addition, they can easily provide the exact coordinates of palm and finger locations, orientation and configurations by using sensors attached to the gloves. However, this approach requires the user to be connected to the computer physically, which blocks the ease of interaction between user and computer. In addition, the price of these devices is quite high. However, the modern glove-based approach uses the technology of touch, which more promising technology and it is considered Industrial-grade haptic technology. Where the glove gives haptic feedback that makes user sense the shape, texture, movement and weight of a virtual object by using microfluidic technology

- *Hand Gestures Based on Computer Vision Approach*

The camera vision-based sensor is a common, suitable and applicable technique because it provides contactless communication between humans and computers. Different configurations of cameras can be utilized, such as monocular, fisheye, TOF and IR. However, this technique involves several challenges, including lighting variation, background issues, the effect of occlusions, complex background, processing time traded against resolution and frame rate and foreground or background objects presenting the same skin color tone or otherwise appearing as hands. These challenges will be discussed in the following sections. A simple diagram of the camera vision-based sensor for extracting and identifying hand gestures is presented in fig.

- *Motion-Based Recognition*

Motion-based recognition can be utilized for detection purposes; it can be extracting the object through a series of image frames. The AdaBoost algorithm utilized for object detection, characterization, movement modelling, and pattern recognition is needed to recognize the gesture. The main issue encounter motion recognition is this is an occasion if one more gesture is active at the recognition process and also dynamic background has a negative effect. In addition, the loss of gesture may be caused by occlusion among tracked hand gesture or error in region extraction from tracked gesture and effect long-distance on the region appearance.

Figure 2.2.4: Motion recognition using frame difference subtraction to extract hand features

Two stages for efficient hand detection were proposed. First, the hand detected for each frame and center point is used for tracking the hand. Then, the second stage matching model applying to each type of gesture using a set of features is extracted from the motion tracking in order to provide better classification where the main drawback of the skin color is affected by lighting variations which lead to detect non-skin color. A standard face detection algorithm and optical flow computation was used to give a user-centric coordinate frame in which motion features were used to recognize gestures for classification purposes using the multiclass boosting algorithm.

A real-time dynamic hand gesture recognition system based on TOF was offered in which motion patterns were detected based on hand gestures received as input depth images. These motion patterns were compared with the hand motion classifications computed from the real dataset videos which do not require the use of a

segmentation algorithm. Where the system provides good result except the depth rang limitation of TOF camera. In YUV color space was used, with the help of the CAM Shift algorithm, to distinguish between background and skin color, and the naïve Bayes classifier was implemented to assist with gesture recognition. The proposed system faces some challenges such as illumination variation where light changes affect the result of the skin segment. Other challenges are the degree of gesture freedom which affect directly on the output result by change rotation. Next, hand position capture problem, if hand appears in the corner of the frame and the dots which must cover the hand does not lie on hand that may led to failing captured user gesture. In addition, the hand size quite differs between humans and maybe causes a problem with the interaction system. However, the major still challenging problem is the skin-like color which affects overall system and can abort the result.

2.3 LITERATURE REVIEW

[1] In this paper which is titled as "vision-Based Hand Gesture Recognition Using Combinational Features" where Yu et al. [1] used YCbCr color model to distinguish skin-colored pixels from the background. The required portion of the hand was extracted using this color model and filtered using median filter and smoothing filter. The edges were detected and features extracted were hand perimeter, aspect ratio, hand area after which Artificial Neural Network (ANN) was used as classifier to recognize a gesture. Accuracy rate obtained was 97.4%.

In this paper, it has presented a method based on real-time hand tracking and gesture recognition from extracted hand features using depth images and contour extraction. The system's performance evaluation results have shown that this efficient interface can be used by the users with ease and accuracy in all lighting conditions. In the future, more applications can be controlled using hand gestures using this method. These applications include software and hardware applications.

This basically presents a feature extraction method for hand gesture based on multi-layer perceptron. The feature of hand skin color in the YCbCr color space is used to detect hand gesture. The hand silhouette and features can be accurately extracted in means of binarizing the hand image and enhancing the contrast. Median and smoothing filters are integrated to remove the noise. Combinational parameters of Hu invariant moment, hand gesture region, and Fourier descriptor are created to form a new feature vector which can recognize hand gesture. To confirm the robustness of this proposed method, a dataset including 3500 images is built. Experimental results demonstrate that our system can successfully recognize hand gesture with 97.4% recognition rate.

Here it is shown the usage of Combinational login. In digital circuit theory, combinational logic (sometimes also referred to as time-independent logic) is a type of digital logic which is implemented by Boolean circuits, where the output is a pure function of the present input only. This is in contrast to sequential logic, in which the output depends not only on the present input but also on the history of the input. In other words, sequential logic has memory while combinational logic does not. An alternate term is combinatorial logic, though this usage may be considered controversial.

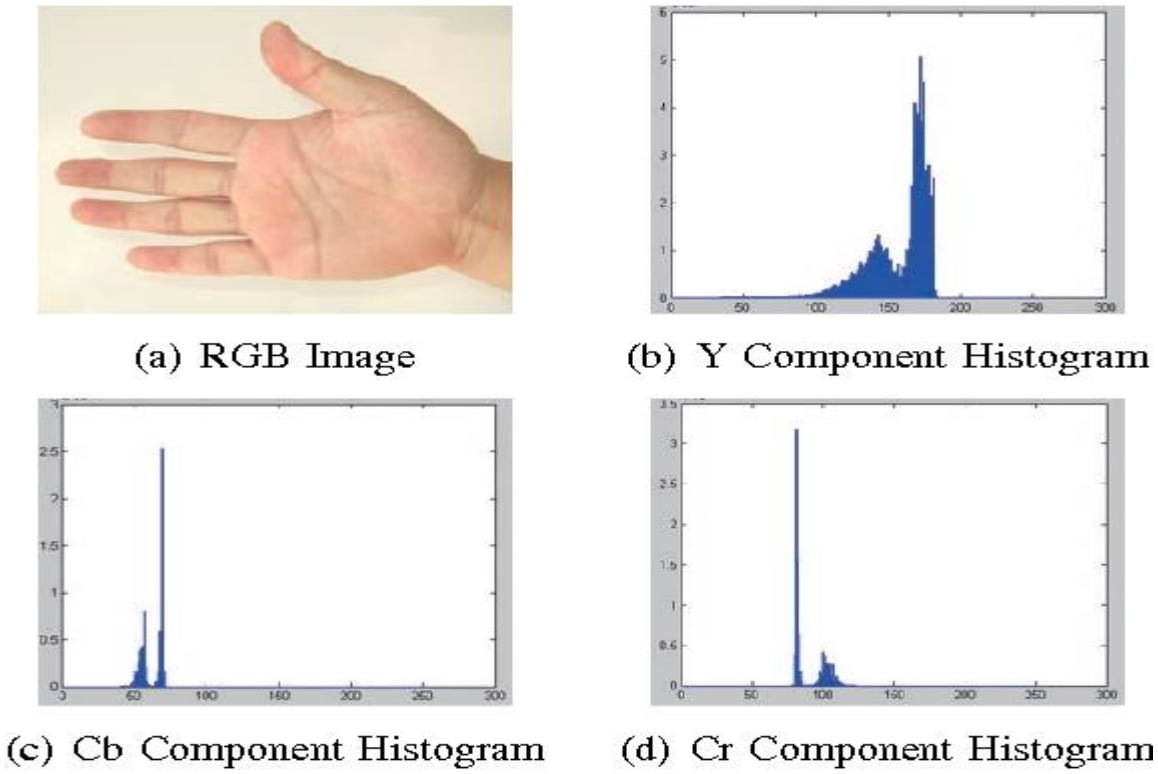

Figure 2.3.1: RGB Image and YCbCr Component Histograms

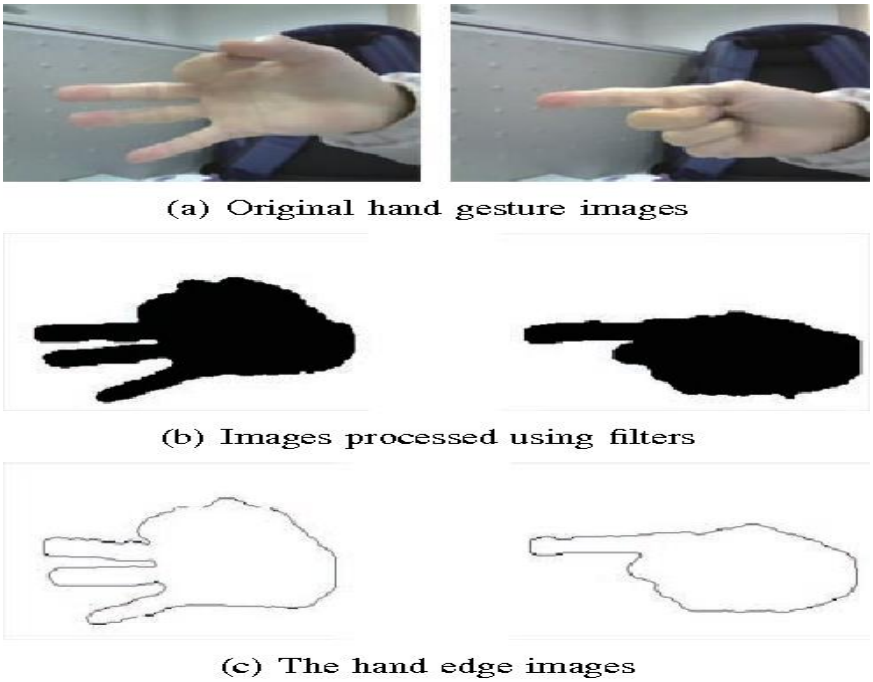

Figure 2.3.2: Original hand gesture images and images processed by filters

[2] In this paper which titled "Vision Based Approach to Sign Language Recognition" it proposed an algorithm for automatically recognizing some certain amount of gestures from hand movements to help deaf and dumb and hard hearing people. Hand gesture recognition is quite a challenging problem in its form. It considered a fixed set of manual commands and a specific environment, and develop a effective, procedure for gesture recognition. The approach contains steps for segmenting the hand region, locating the fingers, and finally classifying the gesture which in general terms means detecting, tracking and recognizing. The algorithm is non-changing to rotations, translations and scale of the hand. It will be demonstrating the effectiveness of the technique on real imagery.

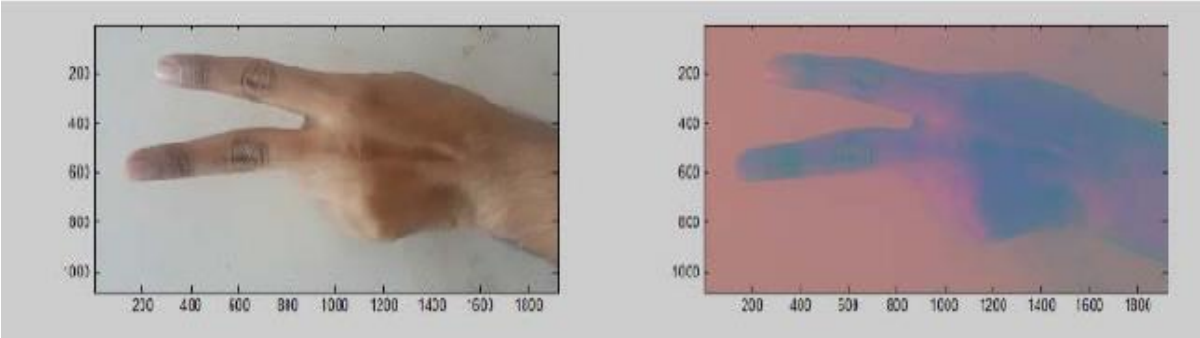

Figure 2.3.3: Converted image from RGB to YCbCr color space

[3] In this paper which titled" Appearance-Based Hand Sign Recognition from Intensity Image Sequences" In this paper, it presents a new approach to recognizing hand signs. In this approach, motion recognition (the hand movement) is tightly coupled with spatial recognition (hand shape). The system uses multiclass, multidimensional discriminant analysis to automatically select the most discriminating linear features for gesture classification. A recursive partition tree approximator is proposed to do classification. This approach combined with the previous work on hand segmentation forms a new framework which addresses the three key aspects of hand sign

interpretation: hand shape, location, and movement. The framework has been tested to recognize 28 different hand signs. The experimental results show that the system achieved a 93.2% recognition rate for test sequences that had not been used in the training phase. It is shown that the approach provide performance better than that of nearest neighbor classification in the eigen subspace.

[4] In this paper which titled "Hand Gesture Recognition for Man-Machine interaction" Many researchers used Vision-based approaches for identifying hand gestures. Kapuscinski et al. found out the skin-colored region from the input image captured and then this image with desired hand region was intensity normalized and histogram was found out for the same. Feature extraction step was performed using Hit-Miss Transform and the gesture was recognized using Hidden Markov Model (HMM). Recognition rate obtained was 98%

This addresses some basic problems of hand gesture recognition. Three steps important in building gestural vision-based interfaces are discussed. The first step is the hand location through detection of skin-colored regions. Representative methods based on 2D color histograms are described and compared. The next step is the hand shape (posture) recognition. An approach that uses the morphological hit-miss operation is presented. Finally, application of hidden Markov models in recognition of dynamic gestures is explained. An experimental system that uses the discussed methods in real time is described and recognition results are presented

[5] In this paper which titled "Indian Sign Language Recognition Using Eigen Value Weighted Euclidean Distance Based Classification Technique" it had proposed a system using Eigen value weighted Euclidean distance as a classification technique for recognition of various Sign Languages of India. The system comprises of four parts: Skin Filtering, Hand Cropping, Feature Extraction and Classification. Twenty-four signs were considered in this paper, each having ten samples, thus a total of two hundred forty images was considered for which recognition rate obtained was 97 percent.

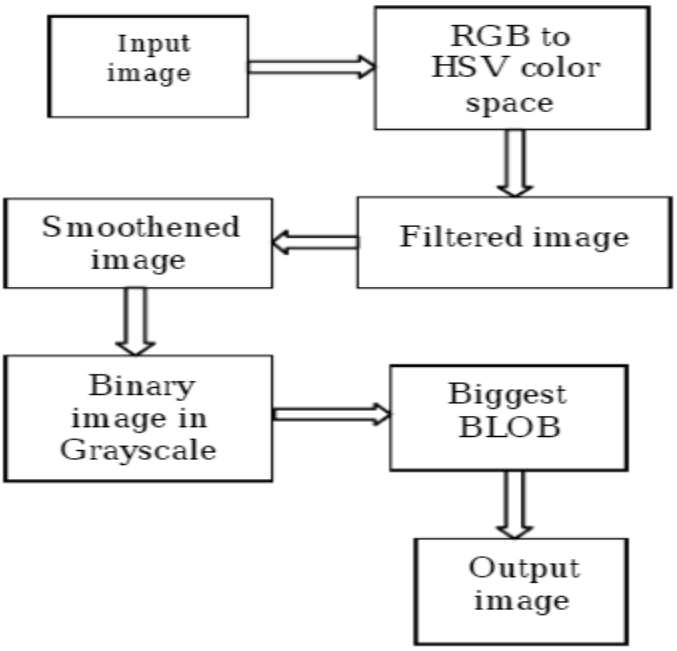

Figure 2.3.4: Block diagram of skin filtering

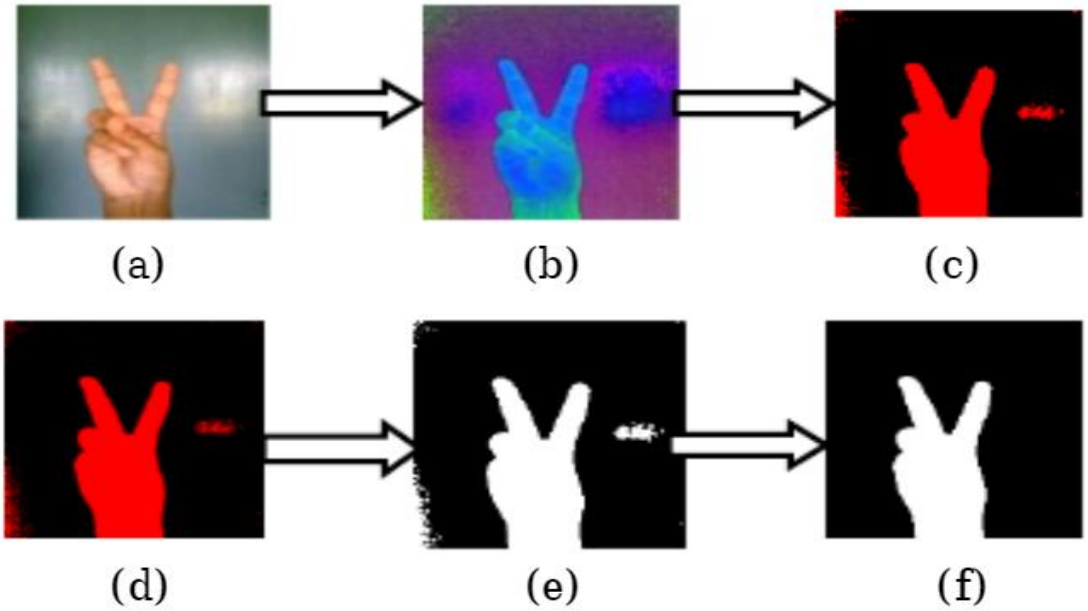

Figure 2.3.5: a) RGB image, b) HSV image, c) Filtered image, d) Smoothened image, e) Binary image in grayscale, f) Biggest BLOB.

The proposed system was implemented with MATLAB version 7.6 (R2008a) and supporting hardware was Intel® Pentium® CPU B950 @ 2.10GHz processor machine, Windows 7 Home basic (64 bit), 4GB RAM and an external 2 MP camera. A system was designed for Indian Sign Language Recognition. It was able to handle different static alphabets of Indian Sign Languages by using Eigen value weighted Euclidean distance between Eigen vectors as a classification technique. It tried to improve the recognition rate compared to the previous works and achieved a success rate of 97%. Moreover, it considered both hands in the paper.

[6] In this paper which titled" A Fast Algorithm for Vision-Based Hand Gesture Recognition for Robot Control" Malima et. used hand gesture recognition for controlling the robot. The Red/Green ratio was computed for determining the skin-colored regions. The center of gravity of the hand was found out along with the farthest distance from it and thus in such a way the fingertips were determined. A circle was made around the center of gravity and number of white pixels beyond that circle was counted to know the desired gesture. Recognition rate obtained was 91%
The paper describes a gesture recognition algorithm which can effectively recognize single-hand gestures in real time under complex environment. The system involves a vocabulary of 20 gestures consisted by part of Chinese sign language letter spelling alphabet and digits. A new complexion model has been proposed to extract hand regions under a variety of lighting conditions. Real-time performance is due to a novel combination of voting theory and relief algorithm. This system is used in our mobile robot control project.

This shows conducting experiments based on images that have acquired using a 4 Mega-Pixel digital camera as well as a simple webcam. The data is collected on uniform as well as cluttered backgrounds. Figure shows a sample result for a hand image displaying the count "two". It shows the output of various stages of our algorithm. Note

that through differencing operations, it obtained the zero-to-one transitions in the 1D signal whose extraction was described in Section 2.5, and is illustrated in Figure 2 as well. The number of these transitions minus one (for the wrist) produces the estimated count Out of 105 samples taken from 21 members of our laboratory, it obtained 96 correct classifications which is approximately 91% of all images used in the experiments. This paper notes that images taken under insufficient light (especially using the webcam) have led to the incorrect results. In these cases the failure mainly stems from the erroneous segmentation of some background portions as the hand region. The algorithm appears to perform well with somewhat complicated backgrounds, as long as there are not too many pixels in the background with skin-like colors.

Chapter-3

COMPONENTS

3.1 PYTHON:

Python is an object-oriented programming language created by Guido Rossum in 1989. Python is a suitable language for fast learning and real-world programming. Python is a powerful high-level programming language.

It is ideally designed for rapid prototyping of complex applications. It has interfaces to many OS system calls and libraries and is extensible to C or C++. Many large companies use the Python programming language including NASA, Google, YouTube, BitTorrent, etc. Python is widely used in Artificial Intelligence, Natural Language Generation, Neural Networks and other advanced fields of Computer Science. Python had a deep focus on code readability. Python language is used by the author due to his cross-platform compatibility as the main coding language for algorithms. Open CV and dlib libraries are integrated in python interpreter for using readymade optimized functions. It can be used for various other technologies like:

- Web Development
- Testing
- Web Scraping
- Data Analysis
- Internet of Things
- Big Data
- Machine Learning
- Computer Graphics

3.2 OpenCV

OpenCV (Open-source Computer Vision) is the Swiss Army knife of computer vision. It has a wide range of modules that can help us with a lot of computer vision problems. But perhaps the most useful part of OpenCV is its architecture and memory management. It provides you with a framework in which you can work with images and video in any way you want, using OpenCV's algorithms or your own, without worrying about allocating and deallocating memory for your images. Open CV libraries and functions are highly optimized and can be used for real time image and video processing. OPENCV's highly optimized image processing functions are used by the author for real time image processing of live video feed from the camera**.** OpenCV's architecture and memory protection areperhaps the most valuable features. The OpenCV supporting tools are :

- IDLE
- Spyder
- PyCharm
- Eric
- Vim
- Atom
- Jupyter
- Anaconda
- PyDev
- Thonny

3.3 NUMPY

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Num array into Numeric, with extensive modifications. NumPy is open-source software and has many contributors.

NumPy targets the Python reference implementation of Python, which is a non-optimizing bytecode interpreter. Mathematical algorithms written for this version of Python often run much slower than compiled equivalents. Python bindings of the widely used computer vision library OpenCV utilize NumPy arrays to store and operate on data. Since images with multiple channels are simply represented as three-dimensional arrays, indexing, slicing or masking with other arrays are very efficient ways to access specific pixels of an image. The NumPy array has universal data structure in OpenCV for images, extracted feature points, filter kernels and many more vastly simplifies the programming workflow and debugging.

Used of NumPy are:

- Airthmetic Operatins
- Statstical Operations
- Bitwise Operations
- Copying and Viewing Arrays
- Stacking
- Matrix Operations

- Searching, Sorting and Counting
- Mathematical Operations
- Broadcasting
- Linear Algebra

3.4 WEBCAM / LAPTOP CAMERA

If we are working in a system containing no camera attached to it(computer), then we need to connect a web camera to it. If we are using a laptop there is no need for using a webcam because most of the laptops are readily available with a camera attached to it. This camera is useful for capturing the video displayed and it will further display the count of fingers or the gestures of the hand as per the given input. In case we are using a web camera a few modifications in the code has to be done and the resolution must be set. A camera loads the OpenCV native library, Instantiates the video capture class, reads the frames, displays both Original and Mask video and then the output.

3.5 MATH LIBRARY

The Python Math Library provides us access to some common math functions and constants in Python, which we can use throughout our code for more complex mathematical computations. The library is a built-in Python module; therefore, you don't have to do any installation to use it. In this article, we will be showing example usage of the Python Math Library is most commonly used functions and constants.

This module provides access to the mathematical functions defined by the C standard.

These functions cannot be used with complex numbers; use the functions of the same name from the cmath module if you require support for complex numbers. The

distinction between functions which support complex numbers and those which don't is made since most users do not want to learn quite as much mathematics as required to understand complex numbers. Receiving an exception instead of a complex result allows earlier detection of the unexpected complex number used as a parameter, so that the programmer can determine how and why it was generated in the first place.

The following functions are provided by this module. Except when explicitly noted otherwise, all return values are floats.

3.6 IMUTILS

Imutils are a series of convenience functions to make basic image processing functions such as translation, rotation, resizing, skeletonization, and displaying Matplotlib images easier with OpenCV and both Python 2.7 and Python 3.

- TRANSLATION - Translation is the shifting of an image in either the *x* or *y* direction. To translate an image in OpenCV you would need to supply the *(x, y)*-shift, denoted as (t_x, t_y) to construct the translation matrix M.

- ROTATION - Rotating an image in OpenCV is accomplished by making a call to cv2.getRotationMatrix2D and cv2.warpAffine. Further care has to be taken to supply the *(x, y)*-coordinate of the point the image is to be rotated about. These calculation calls can quickly add up and make your code bulky and less readable. The rotate function in imutils helps resolve this problem.

- RESIZING - Resizing an image in OpenCV is accomplished by calling the cv2.resize function. However, special care needs to be taken to ensure that

the aspect ratio is maintained. This resize function of imutils maintains the aspect ratio and provides the keyword arguments width and height so the image can be resized to the intended width/height while (1) maintaining aspect ratio and (2) ensuring the dimensions of the image do not have to be explicitly computed by the developer.

- SKELETONIZATION - Skeletonization is the process of constructing the "topological skeleton" of an object in an image, where the object is presumed to be white on a black background. OpenCV does not provide a function to explicitly construct the skeleton, but does provide the morphological and binary functions to do so. For convenience, the skeletonize function of imutils can be used to construct the topological skeleton of the image. The first argument, size is the size of the structuring element kernel. An optional argument, structuring, can be used to control the structuring element -- it defaults to cv2.MORPH-RECT, but can be any valid structuring element.

3.7 OS

The OS module in Python provides a way of using operating system dependent functionality. The functions that the OS module provides allows you to interface with the underlying operating system that Python is running on – be that Windows, Mac or Linux.

The OS module in Python provides functions for interacting with the operating system. OS, comes under Python's standard utility modules. This module provides a portable way of using operating system dependent functionality.

os. system () method executes the command (a string) in a subshell. This method is implemented by calling the Standard C function system (), and has the same limitations. If command generates any output, it is sent to the interpreter standard output stream.

3.8 COPY

The copy() method in Python returns a copy of the Set. We can copy a set to another set using the = operator, however copying a set using = operator means that when we change the new set the copied set will also be changed, if you do not want this behavior then use the copy() method instead of = operator. It also involves deepcopy.

If you change the copied object - you change the original object. deepcopy() creates new objects and does real copying of original objects to new one. deepcopy() copies original object recursively, while copy() create a reference object to first-level data of original object.

Deep copy, refers to a technique by which a copy of an object is created such that it contains copies of both instance members and the objects pointed to by reference members. ... Deep copy is used in scenarios where a new copy (clone) is created without any reference to original data.

Chapter-4

PROPOSED METHOD

The proposed method involves in the easy programming language python and the whole process solely depends on the software components. There is no need for any hardware components unless we don't have a camera attached to the system. And the components are freely available in the internet. We can download and install the components without any investment involved.

The process is as follows. Initially, the packages are to be imported from the library. Then the video capturing is to be done. It converts the video into images and the images are converted from BGR to HSV. The background will be eliminated using "Background subtractor". After that, we defined the skin color range in HSV as two types: UPPER and LOWER range. Masking is done. In masking anything of skin color will be considered as one (or white) and the remaining colors are considered as zero (or black). So, now the computer will have the image of the hand as white and the rest of the background as black in the mask.

Now we will smooth, dilate and blur the image so that any amount of noise present in the system will be further reduced. Then we will find the contours in that image. Contours are the outlines of the area that is being detected. We will find the contour of the maximum area (hand). The rest of the small contours are considered as minimum area (noise). We will approximate the contours to the maximum area to decrease any small amount of noise present. Thus, background elimination will be done. Now we will define the convex hull. A convex hull is a border around the hand which is just like the control that covers the hand. Now we consider the area where the hand is found for the finger counting to take place. And a convex hull is formed around the hand region. Now we will find the defects in the convex hull. Defects are the regions that are not covered by our hand in the convex hull i.e., the distance between finger-to-

finger. It has 3 bonds 1. Start 2. End 3. Far (dot). We have to consider only the distance between fingers but not the other ones i.e., the gap at the end of the palm. So, to exclude them from being considered as defects we've used an angular basis. The proposed method is deeply discussed and justified in the working, code explanation along with a neat flowchart representation. They are discussed in the further process.

4.1 WORKING – CODE EXPLANATION:

- The whole process solely depends on the software components as mentioned above. There is no need for any hardware components. The process is as follows.
- Initially the packages are to be imported from the library.
- Then the video capturing is to be done.
- It converts the video into images and the images are converted from BGR to HSV. Background will be eliminated using “Background subtractor”
- After that we defined the skin color range in HSV as two types: UPPER and LOWER range.
- Masking is done. In masking anything of skin color will be considered as one (or white) and the remaining colors are considered as zero (or black).
- So, now the computer will have the image of the hand as white and the rest of the background as black in the mask.
- Now we will smooth, dilate and blur the image so that any amount of noise present in the system will be further reduced.
- Now we will find the contours in that image.

- Contours are the outlines of the area that is being detected in the region of interest.

- Now we found the contour of the maximum area (hand). The rest of the small contours are considered as minimum area (noise).

- Then we will approximate the contours to maximum area in order to decrease any small amount of noise present. Thus, background elimination will be done.

- Now we will define the convex hull. Convex hull is the border around the hand which is just like the control that covers the hand.

- Now we consider the area as every gesture will have its own unique area ratio which is the basis on which we are differentiating the different hand gestures.

- Now we will find the defects in the convex hull. Defects are the regions that are not covered by our hand in the convex hull i.e., the distance between finger-to-finger. It has 3 bonds 1. Start 2. End 3. Far (dot)

- We have to consider only the distance between fingers but not the other ones i.e., the gap at the end of the palm.

- So, in order to exclude them from being considered as defects we've used angular basis.

- We know that the angle between two fingers is less than 90 degrees. With the help of cosine rule we found the angles.

- So, the regions having an angle in between 0 and 90 are considered as defects and the rest will be ignored. Thus, defect count will be done and it will be displayed as a blue dot.

- Now the rule-based classification will be done to detect finger count.

- The finger count will be equal to the count of number of defects+1.
- For example, consider we've put 3 fingers. Here there will be 2 gaps between 3 fingers. So, the number of defects is 2 and 2+1=3 which gives the count of fingers.
- In case of defects it is divided into 2 categories.
- 1.ANGLE < 90 FOUND: Count is incremented by 1 . If count in this case is greater than zero it returns count+1 and true.
- 2.ANGLE< 90 NOT FOUND: So, there is only one finger and hence there are no defects. So, it returns 0 and false.
- In case of zero, there will be no defects. If the above conditions fail the left over one is b=obviously zero. So, it returns false and zero. So, in the case of true and zero, output is 1, in the case of false and zero output is zero and if not both it prints defect count+1
- Thus, the code works and the input image, HSV converted image and the output image. The finger count will be displayed in the console of python.
- When all the work is done, we've given the escape button as input to close the program running.
- After pressing the escape button all the running code and the background process will be closed.

4.2 BACKGROUNG PROCESSES:

4.2.1 MASKING –

- Image masking is a process of graphics software like Photoshop to hide some portions of an image and to reveal some portions. It is a non-destructive process of image editing.

- Image masking means to apply some images as an mask on the original image or to change the pixel values in the image. To apply a mask on the image, we will use the HoughCircles() method of the OpenCV module. The HoughCircles() method detects the circles in an image.

- When working with data arrays masks can be extremely useful. Masks are an array of Boolean values for which a condition is met (examples below). These Boolean arrays are then used to sort in the original data array (say we only want values above a given value). Here we will use numpy arrays which are especially good for handling data.

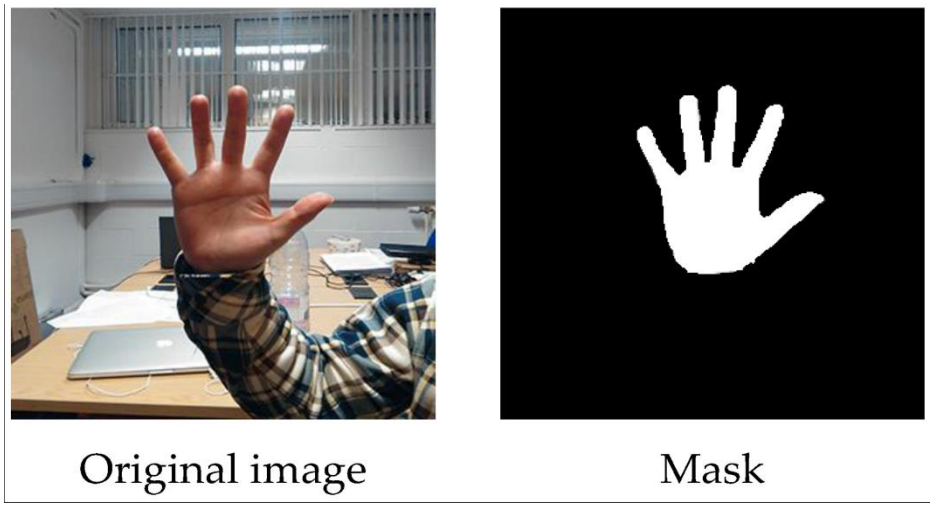

Figure 4.2.1: a)Original image b)Mask image

4.2.2 BACKGROUND SUBTRACTION METHOD –

- Background subtraction is a way of eliminating the background from image. To achieve this, we extract the moving foreground from the static background.

- Background Subtraction has several use cases in everyday life, It is being used for object segmentation, security enhancement, pedestrian tracking, counting the number of visitors, number of vehicles in traffic etc. It is able to learn and identify the foreground mask.

- To remove the background from an image, we will find the contours to detect edges of the main object and create a mask with np. zeros for the background and then combine the mask and the image using the bitwise and operator. In the threshold () method, the last argument defines the style of the threshold.

- In OpenCV we have 3 algorithms to do this operation –

  1.BackgroundSubtractorMOG – It is a Gaussian Mixture-based Background/Foreground Segmentation Algorithm.

  2.BackgroundSubtractorMOG2 – It is also a Gaussian Mixture-based Background/Foreground Segmentation Algorithm. It provides better adaptability to varying scenes due illumination changes etc.

  3.BackgroundSubtractorGMG – This algorithm combines statistical background image estimation and per-pixel Bayesian segmentation.

4.2.3 HSV IMAGE CONVERSION –

- Converting an RGB format Image in an HSV format Image using OpenCV in Python is known as HSV image conversion. An HSV is another type of color space in which H stands for Hue, S stands for Saturation and V stands for Value. A Hue represents color. It is an angle from 0 degrees to 360 degrees.

- A Hue represents color. It is an angle from 0 degrees to 360 degrees.

ANGLE	***COLOR***
0 – 60	***Red***
60 – 120	***Yellow***
120 – 180	***Green***
180 – 240	***Cyan***
240 – 300	***Blue***
300 – 360	***Magenta***

Table 4.2.1: Hue angle and color type

4.2.4 CONVEX HULL –

- A convex hull of an object is the minimum boundary that can completely enclose or wrap the object (or contour of that object).

4.2.5 CONVEXITY DEFECTS –

- Any deviation of the contour from its convex hull is known as the convexity defect.

4.2.6 CONTOUR –

- Contours can be explained simply as a curve joining all the continuous points (along the boundary), having same color or intensity. The contours are a useful tool for shape analysis and object detection and recognition. In OpenCV, finding contours is like finding white object from black background.
- The contour of hand is a series of points which are the boundary pixels of the hand area. After obtaining the contour, the gesture and its shape then can be detected and recognized by using contours analysis. Contour is constituted by the connection of edges, common edge recognition algorithms contain Sobel, Canny, Prewitt, Roberts, and Fuzzy logic methods.

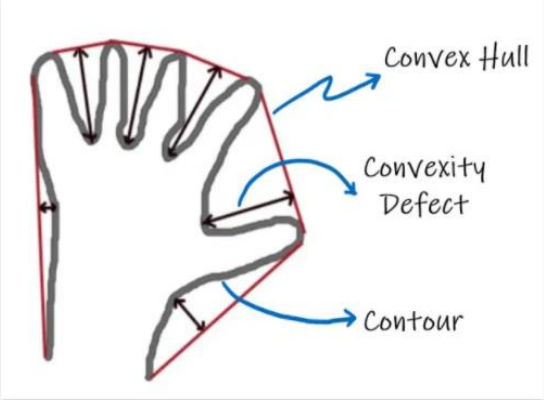

Figure 4.2.2: Figure indication Convex Hull, Convexiuty Defects and Contour

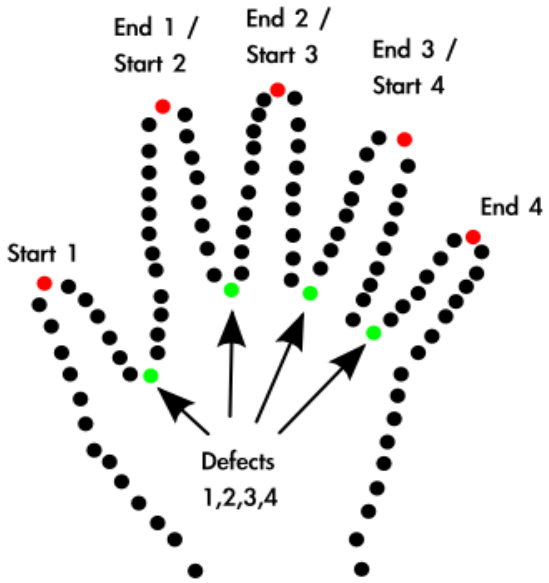

Figure 4.2.3: Figure indicating defects, start and end points of hand detected

4.2.7 SMOOTHING –

Smoothing is used to reduce noise or to produce a less pixelated image. Most smoothing methods are based on low-pass filters, but you can also smooth an image using an average or median value of a group of pixels (a *kernel*) that moves through the image.

4.2.7 DILATION –

Dilation is one of the two basic operators in the area of mathematical morphology, the other being erosion. It is typically applied to binary images, but there are versions that work on grayscale images. The basic effect of the operator on a binary image is to gradually enlarge the boundaries of regions of foreground pixels (*i.e.* white pixels, typically). Thus, areas of foreground pixels grow in size while holes within those regions become smaller.

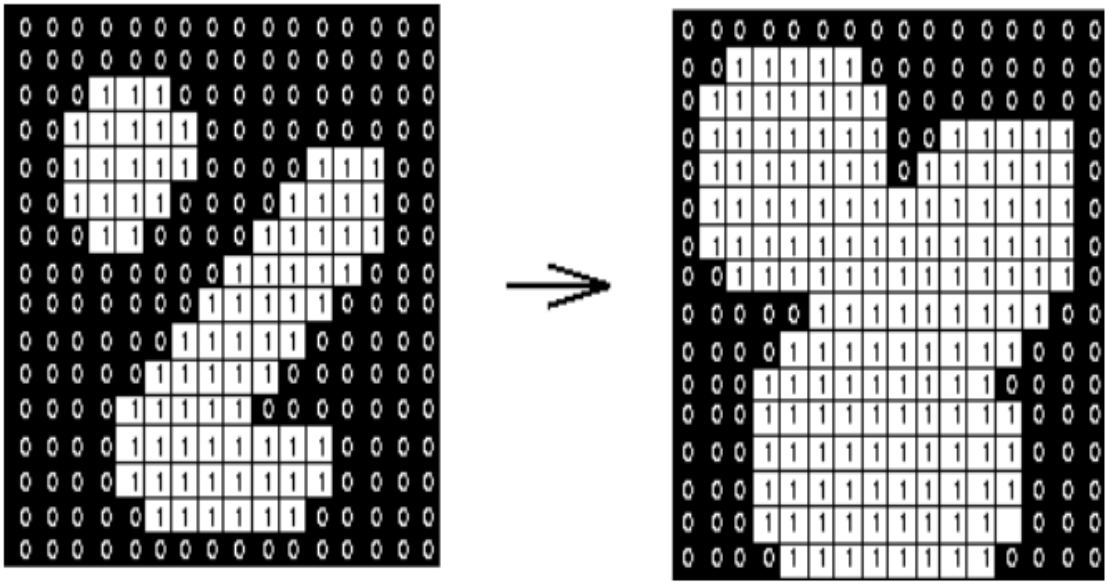

Figure 4.2.4: Effect of dilation using a 3×3 square structuring element

Dilation adds pixels to the boundaries of objects in an image, while erosion removes pixels on object boundaries. The number of pixels added or removed from the objects in an image depends on the size and shape of the structuring element used to process the image.

4.2.8 EROSION –

Erosion (usually represented by ⊖) is one of two fundamental operations (the other being dilation) in morphological image processing from which all other morphological operations are based. It was originally defined for binary images, later being extended to grayscale images, and subsequently to complete lattices. The erosion operation usually uses a structuring element for probing and reducing the shapes contained in the input image.

Erosion is one of the two basic operators in the area of mathematical morphology, the other being dilation. It is typically applied to binary images, but there are versions that

work on grayscale images. The basic effect of the operator on a binary image is to erode away the boundaries of regions of foreground pixels (*i.e.* white pixels, typically). Thus areas of foreground pixels shrink in size, and holes within those areas become larger.

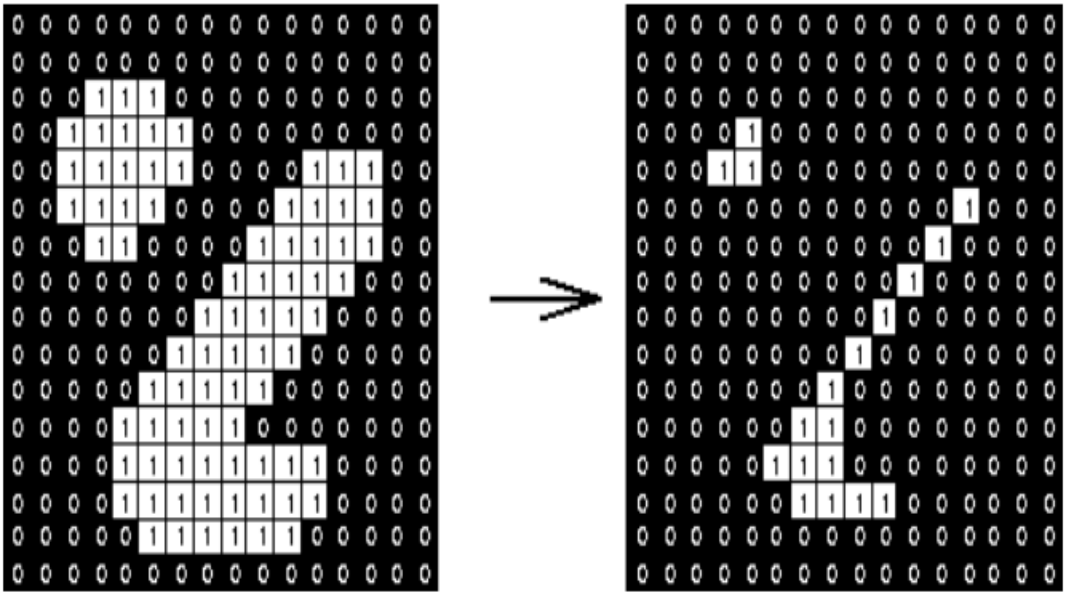

Figure 4.2.5: Effect of erosion using a 3×3 square structuring element

4.3 FLOW CHART:

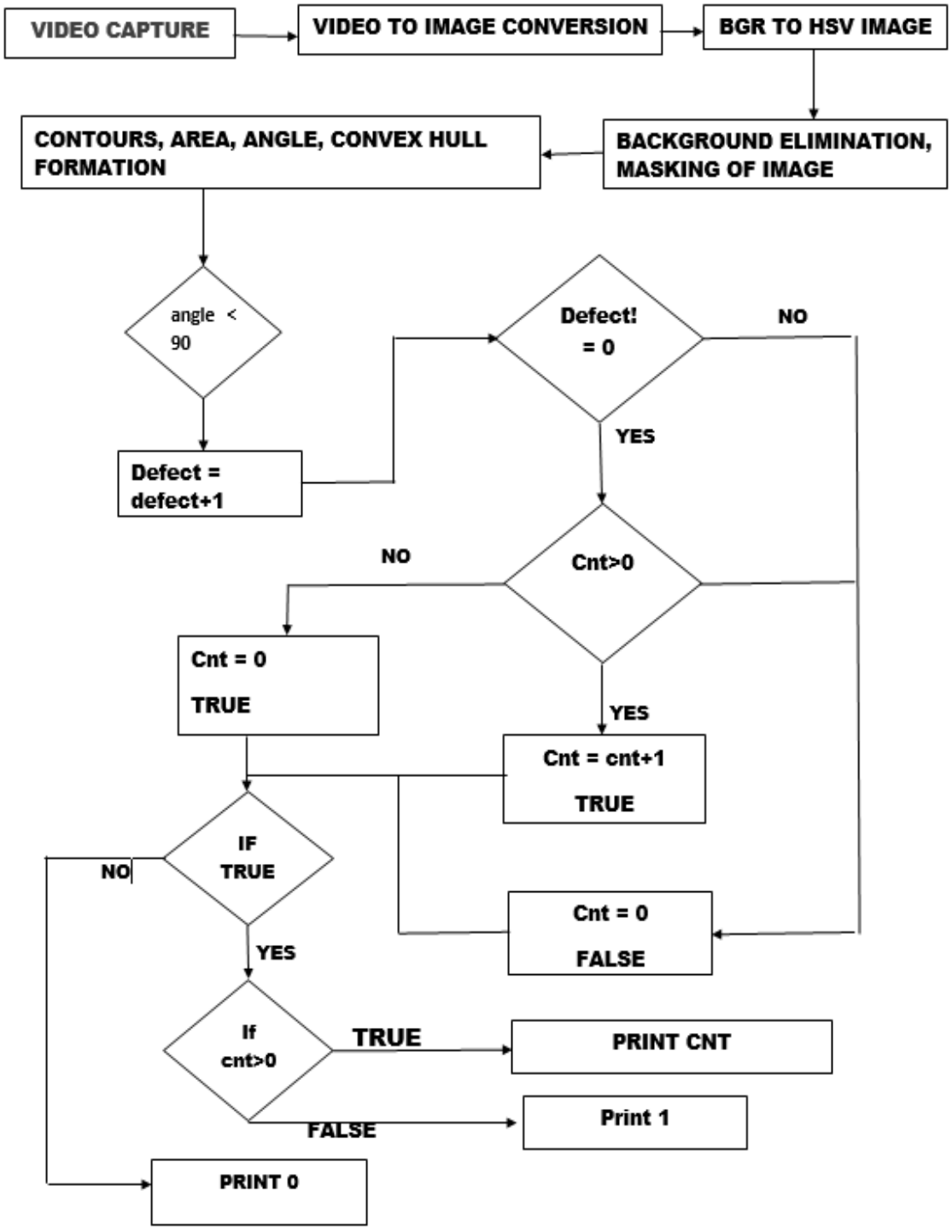

Figure 4.3.1: Flowchart

The above flowchart clearly describes the step-by-step process of the code that will be carried and executed. Initially video capture will be done and the captured video will be converted into images. The captured image which is in the BGR form will be converted into HSV format and the image will undergo a lot of background processes like background elimination, morphological operations like translation, rotation, resization to eliminate the background and to highlight the main object. Now, based on the rule-based classification we find the contours, defects and we draw the convex hull around the hand. The rule-based classification is nothing but angular basis. As we are well aware of the fact that the angle between spacing of fingers is always less than 90 degrees, we find the specific places around the convex hull where the angle is less than 90 degrees. We put a dot i.e., defect in that spot. So, the number of defects incremented by 1 gives the count of the fingers displayed infront of the camera. This process takes place continuously with a time delay of one second. The count i.e., fingers displayed infront of the camera can be viewed in the console of the python.

Chapter-5

RESULTS, DISCUSSIONS AND APPLICATIONS

5.1 RESULTS AND DISCUSSIONS

The output that is the count of the number of fingers detected will be printed in the console of the python in a continuous format and the imaged will be displayed. We will be getting three images they are input image, masked image and the convex hull image in a continuous motion format. The images pattern will be as shown in the below diagrams.

The number of fingers will be displayed in the Python Console shell. The internal process that occurs in the background that are responsible for the successful working of the code are discussed. Image masking is a technique used in graphics software such as Photoshop to cover and remove parts of an image. It is a non-destructive image editing process. Image masking entails using certain images as a mask on the original image or altering the image's pixel values. We'll use the OpenCV module's HoughCircles() method to add a mask to the picture. The HoughCircles() method finds circles in an image. Masks are particularly helpful when dealing with data arrays. A mask is a set of Boolean values that satisfy a condition (examples below). The initial data array is then sorted using these Boolean arrays (say we only want values above a given value). We'll use NumPy arrays instead, which are great for dealing with data.

Fig. 5 gives us an idea about how the outputs are projected.

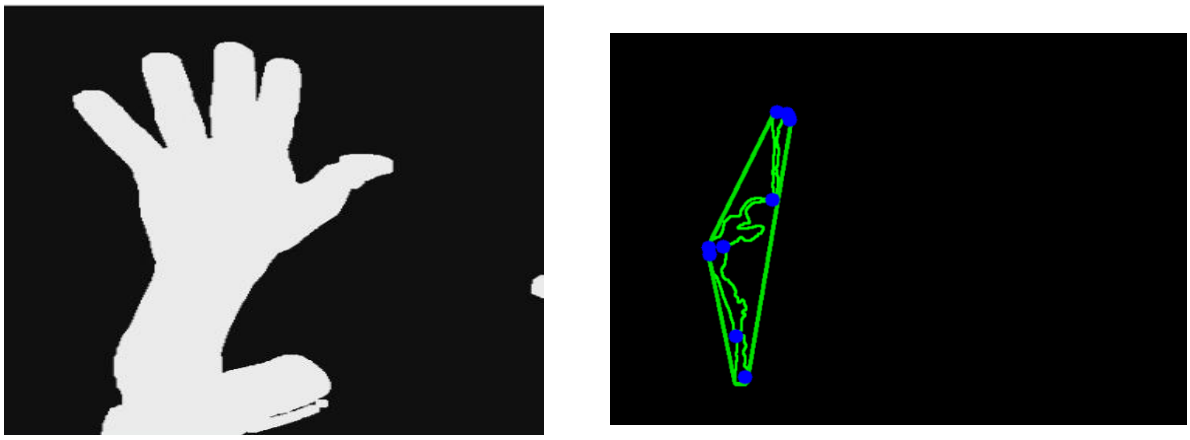

Figure 5.1 a)Mask image b) Hand detected image c)Output image

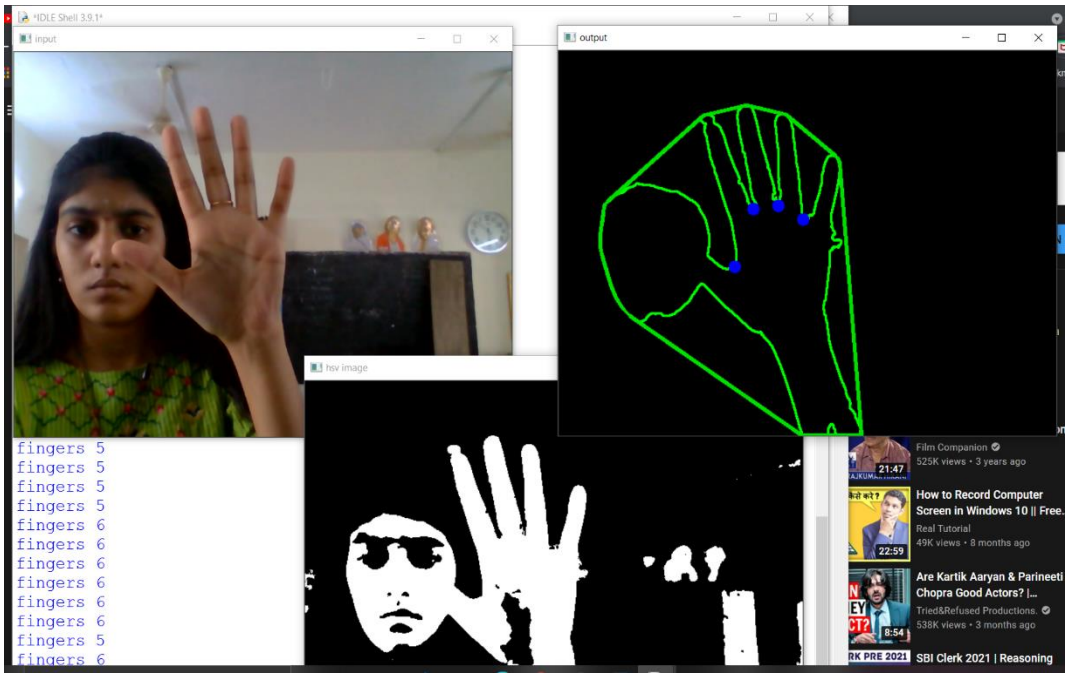

Figure 5.2: Output image for five fingers

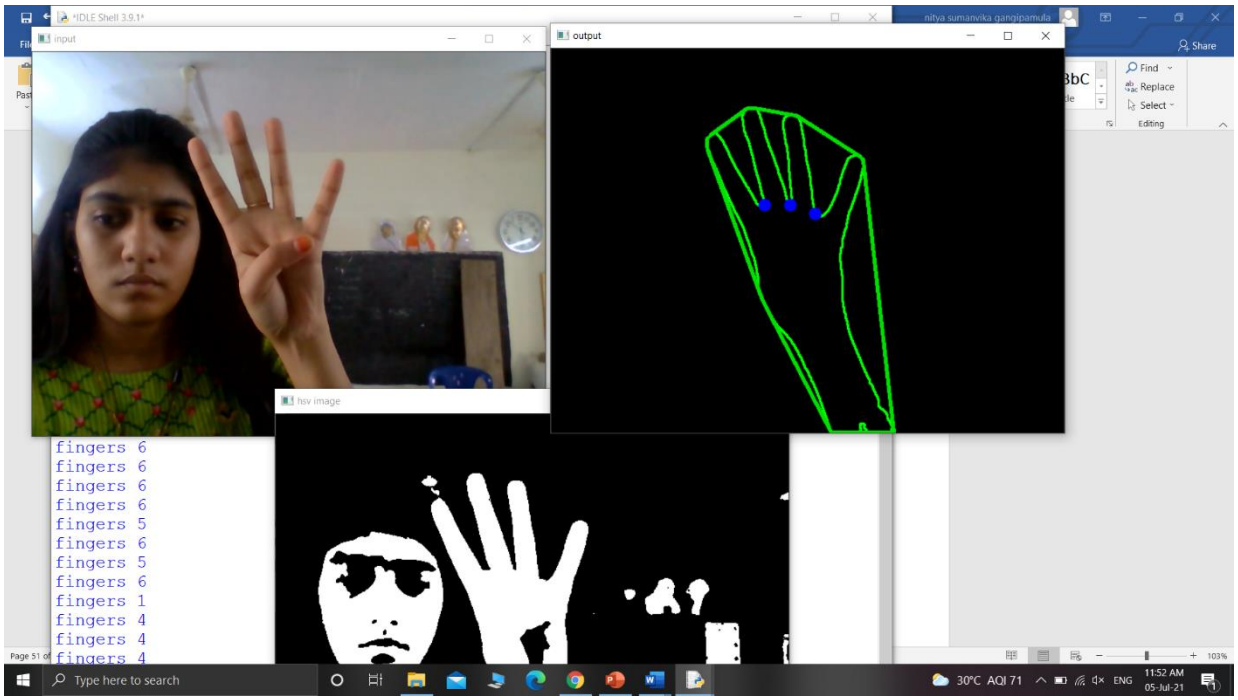

Figure 5.3: Output image for four fingers

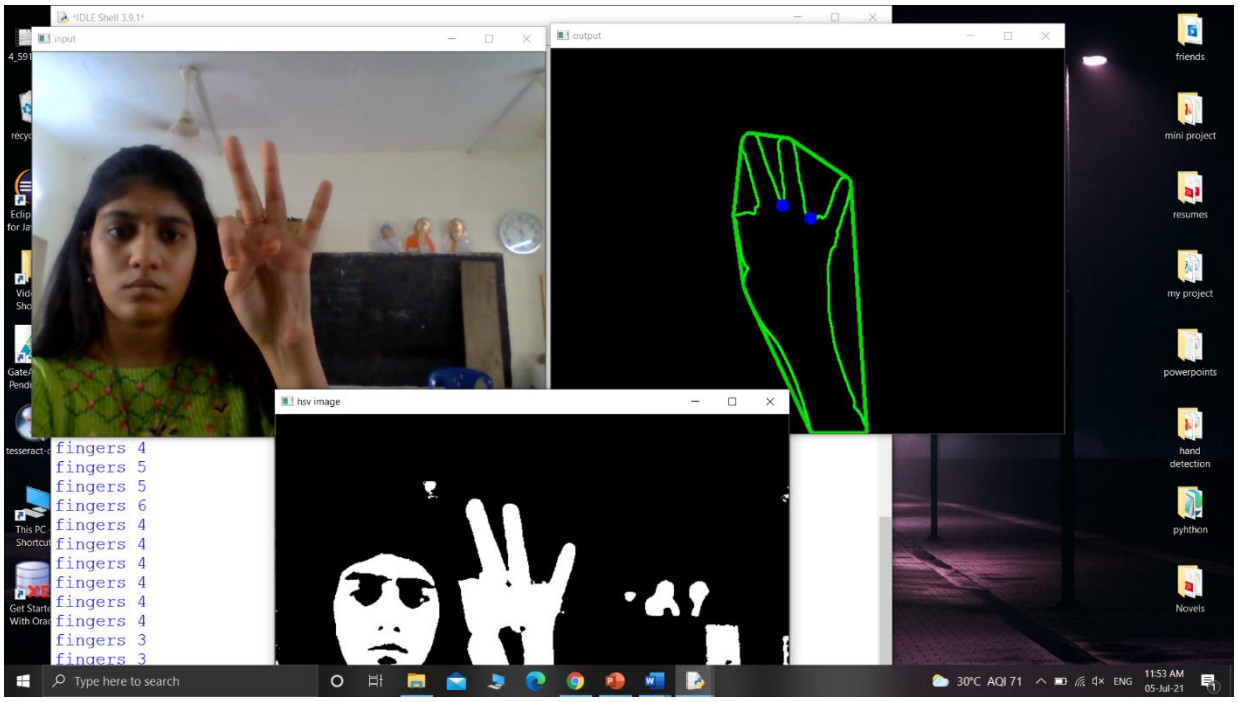

Figure 5.4: Output image for three fingers

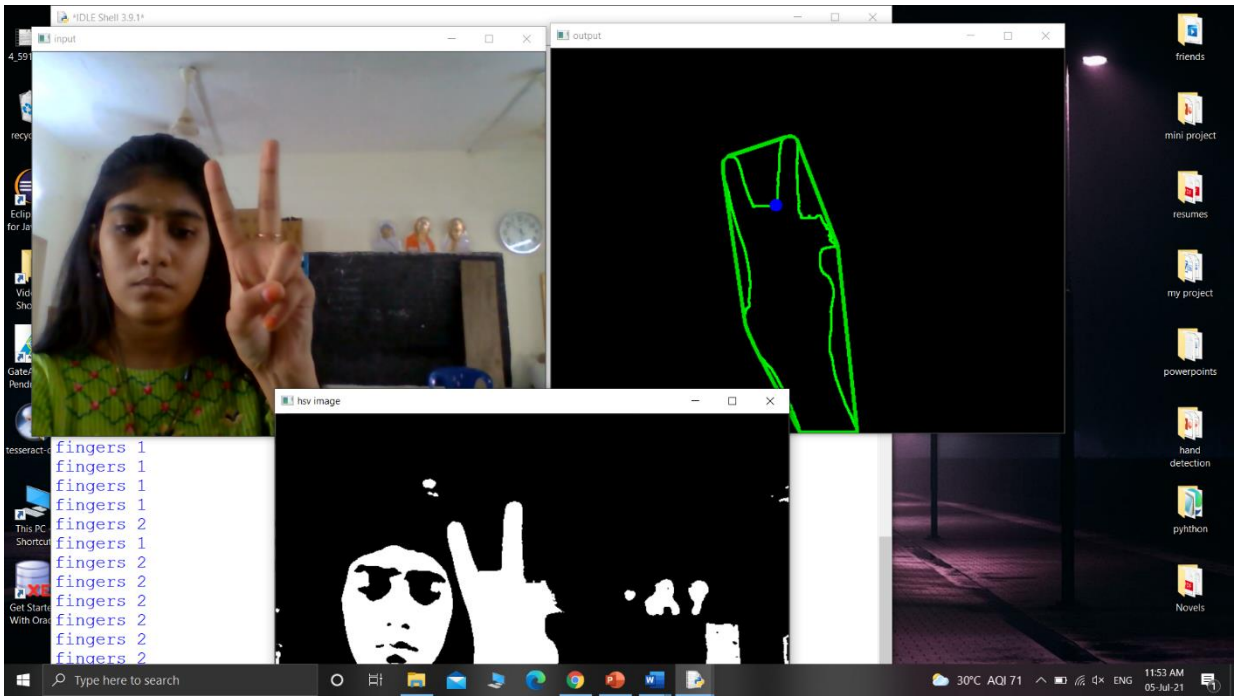

Figure 5.5: Output image for two fingers

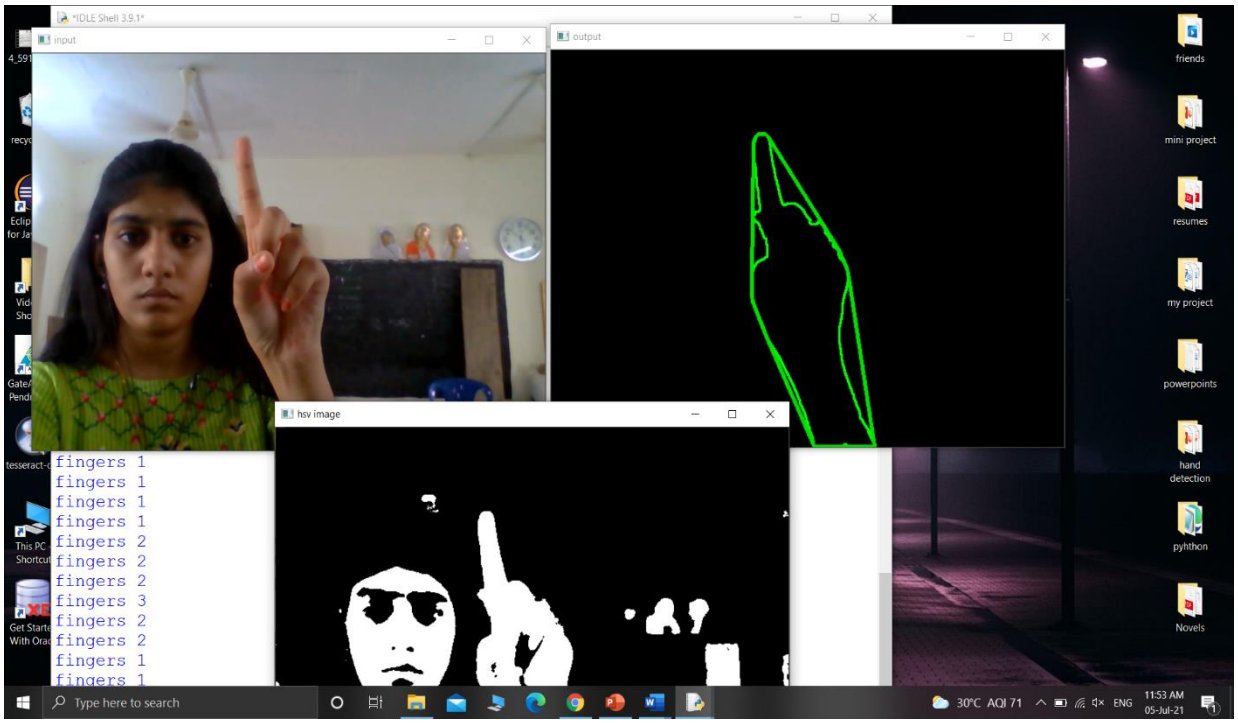

Figure 5.6: Output image for one fingers

The above diagrams represent the outputs that are observed when the hand is detected and the count of fingers are displayed in the console of python continuously. The outputs will be displayed with a delay of 1 second i.e., there is a time gap od 1 second between two consecutive outputs as the delay provided for the detection of pictures and for the display of output is 1 second. We can adjust the time delay as per our convenience.

5.2 APPLICATIONS

- It can be used for both fun and in effective utilization.
- It can be used to educate kids with entertainment.
- It can also be used to send signals.
- One of its applications is to avoid accidents.
- In this a camera is placed in front of the steering. So, when the driver is driving then the hands would be in the steering so he is in safe mode. In case if the camera detected only one hand on the steering, then it checks for the other contours to check if he is talking on his phone or not. If he is talking, then it displays the danger zone and an alarm can be connected to alert him. If both the hands are not on the steering but the driver is present in his seat then it will detect it as a danger zone and an alarm will be switched on making him alert. Moreover, we can also modify the code so as a message can be sent to his friends to make them alert that he is not in a condition to drive thus by making them know about his whereabouts and his condition. Thus, it can be used to avoid accidents.
- It makes our work easy if we have to count fingers in a particular group of people. It counts easily the number of fingers in a group and the error possibility is also less because we know that machine-made work is less prone to errors than that of man-made work.

Chapter-6

CONCLUSIONS AND FUTURE DIRECTIONS

6.1 CONCLUSIONS

Thus by the proposed methodology we can conclude that this is one of the most compatible and easiest way for the hand gesture detection that any individual can use and moreover this method is user friendly as python is readily available in online and it is easily understood for beginners. We can develop it to further many more interesting and useful projects not only for fun but also for life saving and time saving cases. By using this work becomes easy and by further developing it we can create wonders and can also be used to save mankind. It can be implemented in various kinds of fields to make work easy. By using this we can further extend to daily-life applications for safety purposes like to avoid accidents by placing the camera in front of the driver in the car and detecting the gesture and programming it in a way to send a message based on that gesture if he not in a position to drive or control. Not only this we can develop similar projects to save mankind. It Is also used for educational purposes like in the case of teaching children with fun and entertainment. Thus, by using this we can not only count fingers and hand gestures we can develop it to further many more interesting and useful projects not only for fun but also for life-saving and time-saving cases.

6.2 FUTURE DIRECTIONS

- By using this work becomes easy and by further developing it we can create wonders and can also be used to save mankind.
- It can be implemented in various kinds of fields to make work easy.

- By using this we can avoid accidents like in case of placing the camera in front of the driver in the car.
- Not only this we can develop similar projects to save mankind.
- It Is also used for educational purposes like in case of teaching children with fun and entertainment.

REFERENCES

[1] C. Yu, X. Wang, H. Huang, J. Shen and K. Wu, “Vision-Based Hand Gesture Recognition Using Combinational Features”, IEEE Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, 2010, pp. 543-546

[2] Gupta, H., Ramjiwal, A., & T. Jose, J. (2018). Vision Based Approach to Sign Language Recognition. International Journal of Advances in Applied Sciences, 7(2), 156. https://doi.org/10.11591/ijaas.v7.i2.pp156-161

[3]Cui, Y., & Weng, J. (2000). Appearance-Based Hand Sign Recognition from Intensity Image Sequences. Computer Vision and Image Understanding, 78(2), 157–176. https://doi.org/10.1006/cviu.2000.0837

[4] T. Kapuscinski and M. Wysocki, “Hand Gesture Recognition for Man-Machine interaction”, Second Workshop on Robot Motion and Control, October 18-20, 2001, pp. 91-96

[5] Singha, Joyeeta & Das, Karen. (2013). Indian Sign Language Recognition Using Eigen Value Weighted Euclidean Distance Based Classification Technique. International Journal of Advanced Computer Science and Applications. 4. 10.14569/IJACSA.2013.040228.

[6] A. Malima, E. Ozgur and M. Cetin, "A Fast Algorithm for VisionBased Hand Gesture Recognition for Robot Control," 2006 IEEE 14th Signal Processing and Communications Applications, Antalya, 2006, pp. 1-4

[7] R. Rosales, V. Athitsos, L. Sigal, and S. Sclaroff, "3D Hand Pose Reconstruction Using Specialized Mappings", IEEE International Con. on Computer Vision, pp. 378-385,2001

[8] C.-C. Chang, I.-Y Chen, and Y.-S. Huang, "Hand Pose Recognition Using Curvature Scale Space", IEEE International Conference on Pattern Recognition, 2002.

[9] H.J. Lee and J.H. Chung, "Hand Gesture Recognition Using Orientation Histogram," in Proceedings of the IEEE Region 10 Conference (TENCON 99), Vol., 2, pp. 1355-1358, 1999.385, 2001

[10] S. S. Ge, Y. Yang, T. H. Lee. Hand Gesture Recognition and Tracking Based on Distributed Locally Linear Embedding [J]. Image and Vision Computing, 2008, Vol. 26(12): 1607-1620.

[11] Q. Chen, N. D. Georganas. Hand Gesture Recognition Using Haar-like Features and a Stochastic Context-free Grammar [J]. IEEE Transactions on Instrumentation and Measurement, 2008, Vol. 57(8): 1563-1571.

[12] B. Tusor, A. R. Varkonyi-Koczy. Circular Fuzzy Neural Network Based Hand Gesture and Posture Modeling [C]. Instrumentation and Measurement Technology Conference (I2MTC), 2010: 815-820.

[13] A. Utsumi. J. Ohya. Image Segmentation for Human Tracking Using Sequential-image Based Hierarchical Adaptation [C]. Proceeding of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1998: 911-916.

[14] J. C. Harsanyi, C. I. Chang. Hyperspectral Image Classification and Dimensionality Reduction: an Orthogonal Subspace Projection Approach [J]. IEEE Transactions on Geoscience and Remote Sensing, 1994, Vol. 32(4): 779-785.

[15] H. Zhou, D. Lin. Static Hand Gesture Recognition Based on Local Orientation Histogram Feature Distribution Model [C]. Proceedings of the Conference on Computer Vision and Pattern Recognition Workshop, 2004. Vol. 10.

[16] A. A. Argyros, M. I. A. Lourakis. Vision-based Interpretation of Hand Gestures for Remote Control of a Computer Mouse [C]. Proceeding of the International Conference on Computer Vision in Human-computer Interaction, 2006: 40-51.

[17] T. Lee, T. Hollerer. Handy AR: Markerless Inspection of Augmented Reality Objects Using Fingertip Tracking [C]. 11th IEEE International Symposium on Wearable Computers, 2007: 83-90.

[18] W. T. Freeman, M. Roth. Orientation Histograms for Hand Gesture Recognition [C]. IEEE International Workshop on Automatic Face and Gesture Recognition, 1995.

APPENDIX

```
import cv2

import numpy as np

import imutils

import os

import copy

import math


def calcfing (res, draw):

   hull = cv2.convexHull(res, returnPoints = False)

   if len (hull)>3:

      defects = cv2.convexityDefects(res, hull)

      if defects is not None:

         cnt = 0

         for i in range(defects.shape[0]):

            s, e, f, d = defects[i][0]

            start = tuple(res[s][0])

            end = tuple(res[e][0])

            far = tuple(res[f][0])

            a = math.sqrt((end[0]-start[0]) **2+ (end[1]-start[1]) **2)

            b = math.sqrt((end[0]-start[0]) **2+ (end[1]-start[1]) **2)

            c = math.sqrt((end[0]-start[0]) **2+ (end[1]-start[1]) **2)
```

```
            angle = math.acos((b ** 2 + c ** 2 - a**2) / (2 * b * c))
            if angle <=math.pi/2:
                cnt = cnt+1
                cv2.circle(draw, far, 8, [255,0,0], -1)
        if cnt>0:
            return True, cnt + 1
        else:
            return True, 0
    return False, 0
cam = cv2.VideoCapture(0)

while cam.isOpened():
    ret, image = cam.read()
    image = cv2.bilateralFilter(image,5,50,100)
    image = cv2.flip(image,1)
    cv2.imshow('input', image)
    bgModel = cv2.createBackgroundSubtractorMOG2(0,50) #here 50 is sharpness
    fgmask = bgModel.apply(image)
    kernel = np.ones((3,3), np.uint8)
    fgmask = cv2.erode(fgmask, kernel, iterations = 1)
    img = cv2.bitwise_and(image, image, mask = fgmask)

    hsv = cv2.cvtColor(img,cv2.COLOR_BGR2HSV)
```

```
l = np.array([0,48,80], dtype ="uint8")
u = np.array([255,255,255], dtype="uint8")
skin = cv2.inRange(hsv, l, u)
cv2.imshow("hsv image", skin)


hand = copy.deepcopy(skin)
contours, hierarchy = cv2.findContours(hand, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
length = len(contours)
if length>0:
    for i in range(length):
        area = cv2.contourArea(contours[i])
        if area>-1:
            ci = i
            res = contours[ci]
    hull = cv2.convexHull(res)
    draw = np.zeros(img.shape, np.uint8)
    cv2.drawContours(draw, [res], 0, (0,255,0),2)
    cv2.drawContours(draw, [hull], 0, (0,225,0),3)
    isFinishCal, cnt = calcfing(res, draw)
    print ("fingers", cnt)
    cv2.imshow('output', draw)
    k = cv2.waitKey(10)
if k == 27:
```

```
break;
```

Student Details

1. Roll number : 17A91A04D1
Name of the Student : Nitya Sumanvika Gangipamula
Phone Number : 8919204018
Mail id : sumanvika.02@gmail.com

2. Roll number : 17A91A04E4
Name of the Student : Surya Deepthi Kumpatla
Phone Number : 9182896344
Mail id : deepthikumpatla@gmail.com

3. Roll number : 17A91A04G5
Name of the Student : Manasa Seepana
Phone Number : 7893781814
Mail id : manasa2042000@gmail.com

4. Roll number : 17A91A04D5
Name of the Student : Veera Avinash Gudhe
Phone Number : 8247686404
Mail id : veeraavinash123@gmail.com

5. Roll number : 17A91A04C9
Name of the Student : Joy Kishore Gadipena
Phone Number : 9866390063
Mail id : Joypaulvijayaraj@gmail.com