VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belgaum-590018

A PROJECT REPORT **(15CSP85)** ON

"RECOGNITION OF HAND GESTURES OF HUMANS USING MACHINE LEARNING"

Submitted in Partial fulfillment of the Requirements for the Degree of

Bachelor of Engineering in Computer Science & Engineering

By

AMAN GUPTA (1CR16CS186)

AMIRUL HAQUE (1CR16CS015)

GOVINDA KUMAR GUPTA (1CR16CS053)

ISHAN MISHRA (1CR16CS057)

Under the Guidance of,

MRs. GOPIKA D.

Assistant Professor, Dept. of CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR INSTITUTE OF TECHNOLOGY

#132, AECS LAYOUT, IT PARK ROAD, KUNDALAHALLI, BANGALORE-560037

CMR INSTITUTE OF TECHNOLOGY

#132, AECS LAYOUT, IT PARK ROAD, KUNDALAHALLI, BANGALORE-560037

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the project work entitled **"REGONITION OF HAND GESTURES OF HUMANS USING MACHINE LEARNING"** carried out by **Mr. AMAN GUPTA**, USN **1CR16CS186**, **Mr. AMIRUL HAQUE**, USN **1CR16CS015**, **Mr. GOVINDA KUMAR GUPTA**, USN **1CR16CS053, Mr. ISHAN MISHRA**, USN **1CR16CS057,** bonafide students of CMR Institute of Technology, in partial fulfillment for the award of **Bachelor of Engineering** in Computer Science and Engineering of the Visveswaraiah Technological University, Belgaum during the year 2019-2020. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library.

The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.

________________	________________	________________
Mrs. Gopika D.	**Dr. Prem Kumar Ramesh**	**Dr. Sanjay Jain**
Assistant Professor	**Professor & Head**	**Principal**
Dept. of CSE, CMRIT	**Dept. of CSE, CMRIT**	**CMRIT**

External Viva

Name of the examiners Signature with date

1. ________________
2. ________________

DECLARATION

We, the students of Computer Science and Engineering, CMR Institute of Technology, Bangalore declare that the work entitled "**Recognition of Hand Gesture of Humans using Machine Learning**" has been successfully completed under the guidance of **Mrs. Gopika D. , Assistant Professor**, Computer Science and Engineering Department, CMR Institute of technology, Bangalore. This dissertation work is submitted in partial fulfillment of the requirements for the award of Degree of Bachelor of Engineering in Computer Science and Engineering during the academic year 2019 - 2020. Further the matter embodied in the project report has not been submitted previously by anybody for the award of any degree or diploma to any university.

Place: Bangalore

Date: 17-Jun-2020

Team members:

AMAN GUPTA (1CR16CS186)

AMIRUL HAQUE (1CR16CS015)

GOVINDA KUMAR GUPTA(1CR16CS053)

ISHAN MISHRA(1CR16CS057)

ABSTRACT

The goal for the project was to develop a new type of Human Computer Interaction system that subdues the problems that users have been facing with the current system. The project is implemented on a Linux system but could be implemented on a windows system by downloading some modules for python. The algorithm applied is resistant to change in background image as it is not based on background image subtraction and is not programmed for a specific hand type; the algorithm used can process different hand types, recognizes no of fingers, and can carry out tasks as per requirement. As it is stated within this project report, the main goals were reached. The application is capable of the gesture recognition in real-time. There are some limitations, which we still have to be overcome in future. Hand gesture recognition system received great attention in the recent few years because of its manifoldness applications and the ability to interact with machine efficiently through human computer interaction.

Hand gestures are powerful human to human communication channel which convey a major part of information transfer in our everyday life. Hand gestures are the natural way of interactions when one person is communicating with one another and therefore hand movements can be treated as a non verbal form of communication. Hand gesture recognition is a process of understanding and classifying meaningful movements by the human hands.

ACKNOWLEDGEMENT

We take this opportunity to express our sincere gratitude and respect to **CMR Institute of Technology, Bengaluru** for providing us a platform to pursue our studies and carry out our final year project.

We take great pleasure in expressing our deep sense of gratitude to **Dr. Sanjay Jain,** Principal, CMRIT, Bangalore for his constant encouragement.

We would like to thank **Dr. Prem Kumar ,** Professor and Head, Department of Computer Science &Engineering, CMRIT, Bangalore, who has been a constant support and encouragement throughout the course of this project.

We express our sincere gratitude and we are greatly indebted to **Mrs Sherly Noel, Assistant Professor,** Department of Computer Science & Engineering, CMRIT, Bangalore, for her invaluable co-operation and guidance at each point in the project without whom quick progression in our project was not possible.

We are also deeply thankful to our project guide **Mrs. Gopika D. , Assistant Professor,** Department of Computer Science & Engineering, CMRIT, Bangalore, for critically evaluating our each step in the development of this project and provided valuable guidance through our mistakes.

We also extend our thanks to all the faculty of Computer Science & Engineering who directly or indirectly encouraged us.

Finally, we would like to thank our parents and friends for all their moral support they have given us during the completion of this work.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

In today‘s world, the computers have become an important aspect of life and are used in various fields however, the systems and methods that we use to interact with computers are outdated and have various issues, which we will discuss a little later in this paper. Hence, a very new field trying to overcome these issues has emerged namely HUMAN COMPUTER INTERACTIONS (HCI). Although, computers have made numerous advancement in both fields of Software and Hardware, Still the basic way in which Humans interact with computers remains the same, using basic pointing device (mouse) and Keyboard or advanced Voice Recognition System, or maybe Natural Language processing in really advanced cases to make this communication more human and easy for us. Our proposed project is the Hand gestures recognition system to replace the basic pointing devices used in computer systems to reduce the limitations that stay due to the legacy systems such as mouse and Touchpad. The proposed system uses hand gesture, mostly no of fingers raised within the region of Interest to perform various operations such as Play, Pause, seek forward, seek back word in video player (for instance VLC media player). A static control board restrains the versatility of client and limits the capacity of the client like a remote can be lost, dropped or broken while, the physical nearness of client is required at sight of activity and that is a limitation of the user. The proposed system can be used to control various soft panels like HMI systems, Robotics Systems, Telecommunication System, using hand gestures with help of programming by within python using pyautogui module to facilitate interaction within different functions of computer through the Camera to capture video frames. A Hand Gesture Recognition System recognizes the Shapes and or orientation depending on implementation to task the system into performing some job. Gestures is a form of nonverbal information. A person can make numerous gestures at a time. As humans through vision perceive human gestures and for computer we need a camera, it is a subject of great interest for computer vision researchers such as performing an action based on gestures of the person.

1.1 Digital Image Processing

Image processing is reckon as one of the most rapidly involving fields of software industry with growing applications in all area of work. It holds the possibility of developing ultimate machine in future, which would be able to perform the visual function of living beings. As such, it forms the basis of all kinds of visual automation.

1.2 Hand Gesture Detection and Recognition

A Hand Gesture Recognition System recognizes the Shapes and or orientation depending on implementation to task the system into performing some job. Gestures is a form of nonverbal information. A person can make numerous gestures at a time. As humans through vision perceive human gestures and for computer we need a camera, it is a subject of great interest for computer vision researchers such as performing an action based on gestures of the person.

1.2.1 Detection

Hand detection is relate to the location of the presence of a hand in a still image or sequence of images i.e. moving images. In case of moving sequence, it can be followed by tracking of the hand in the scene but this is more relevant to the applications such as sign language. The underlying concept of hand detection is that human eyes can detect objects, which machines cannot, with that much accuracy as that of humans. From a machine point of view it is just like a man fumble with his senses to find an object.

The factors, which make the hand detection task difficult to solve are:

Variations in image plane and pose

The hands in the image vary due to rotation, translation and scaling of the camera pose or the hand itself. The rotation can be both in and out of the plane.

Skin Colour and Other Structure Components

The appearance of a hand is largely affect by skin colour, size and the presence or absence of additional features like hairs on the hand further added to variability.

Lighting Condition and Background

As shown in Figure 1.1 light source properties affect the appearance of the hand. In addition, the background, which defines the profile of the hand, is important and cannot be ignored.

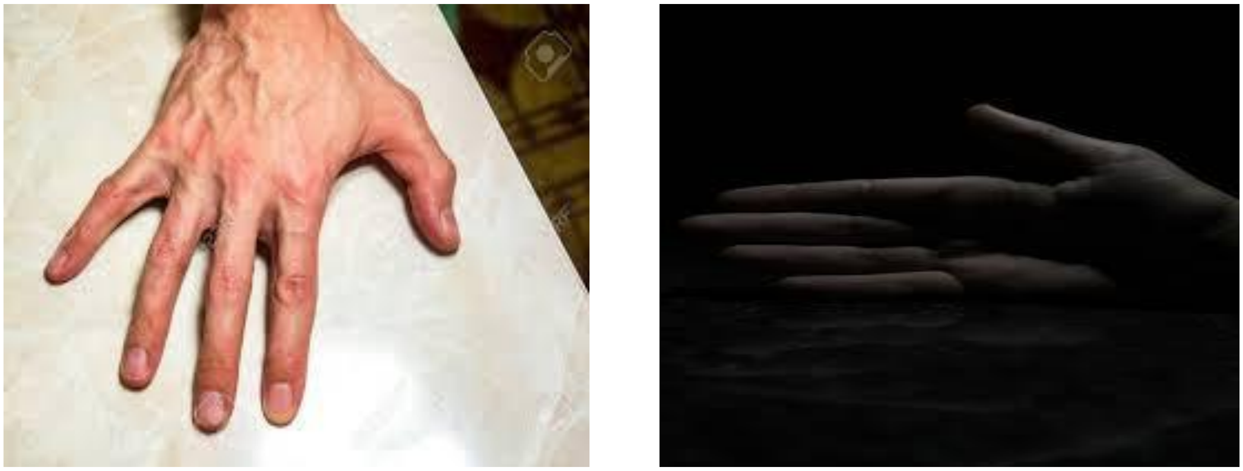


Figure 1.1: Lighting Condition and Background

1.2.2 Recognition

Hand detection and recognition has been significant subject in the field of computer vision and image processing in the past 30 years. There have been considerable achievement and numerous approach develop in this field. **Gesture recognition** is a topic in computer science and language technology with the goal of interpreting human gestures via mathematical algorithms. Many approaches have been made using cameras and computer vision algorithms to interpret sign language. However, the identification and recognition of posture, gait, proxemics, and human behaviours is also the subject of gesture recognition techniques. However, the typical approach of a recognition system has been shown in the below figure:

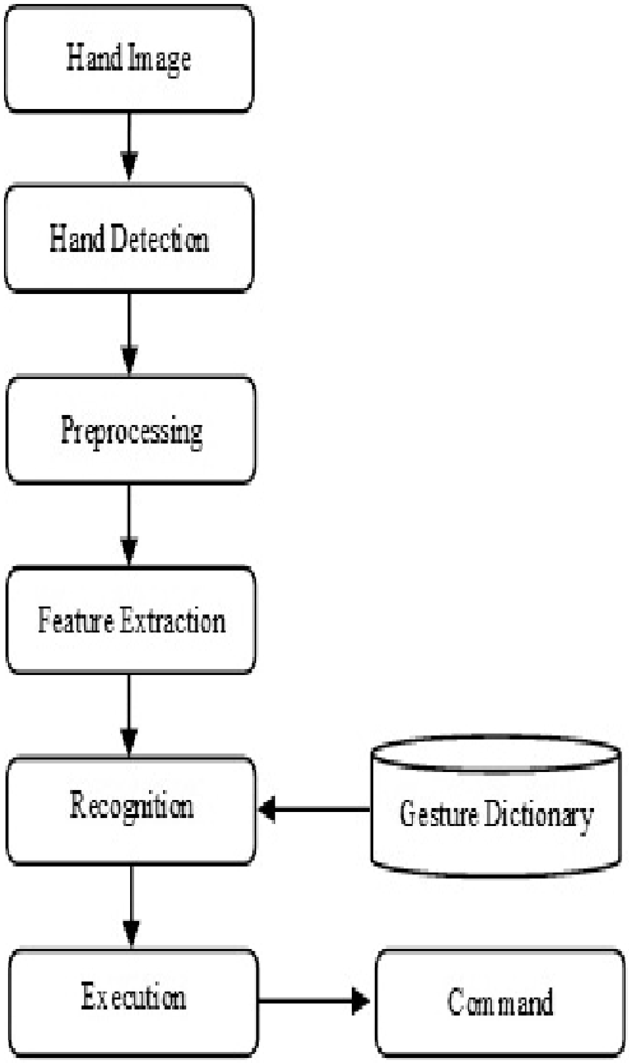


Figure 1.2: Hand Gesture Recognition Flow Chart

1.3 Objective

The objectives of the project are:

1) Study and apply the needed tools, namely:

a) A Media Player like VLC Media Player.

b) The Open CV Computer Vision Library.

c) Algorithms for computer vision and machine learning.

2) Develop a computer vision application for simple gesture recognition.

3) Test the computer application.

4) Document the results of the project.

1.4 Scope

The scope of our project is to develop a realtime gesture recognition system which ultimately controls the media player(i.e. VLC Media Player). During the project, four gestures were chosen to represent four navigational commands for the media player, namely Move Forward, Move Backword, Play, and Stop. A simple computer vision application was written for the detection and recognition of the four gestures and their translation into the corresponding commands for the media player. The appropriate OpenCV functions and image processing algorithms for the detection and interpretation of the gestures were used. Thereafter, the program was tested on a webcam with actual hand gestures in real-time and the results were observed.

CHAPTER 2

LITERATURE SURVEY

2.1 Computer Vision and Digital Image Processing

The sense of sight is arguably the most important of man's five senses. It provides a huge amount of information about the world that is rich in detail and delivered at the speed of light. However, human vision is not without its limitations, both physical and psychological. Through digital imaging technology in [1] and computers, man has transcending many visual limitations. He can see into far galaxies, the microscopic world, the sub-atomic world, and even “observe” infrared, x-ray, ultraviolet and other spectra for medical diagnosis, meteorology, surveillance, and military uses, all with great success.

While computers have been central to this success, for the most part man is the sole interpreter of all the digital data. For a long time, the central question has been whether computers can be design to analyse and acquire information from images autonomously in the same natural way humans can. According to Gonzales and Woods in [2], this is the province of computer vision, which is that branch of artificial intelligence that ultimately aims to “use computers to emulate human vision, including learning and being able to make inferences and taking actions based on visual inputs.”

The main difficulty for computer vision as a relatively young discipline is the current lack of a final scientific paradigm or model for human intelligence and human vision itself on which to build a infrastructure for computer or machine learning. The use of images has an obvious drawback. Humans perceive the world in 3D, but current visual sensors like cameras capture the world in 2D images. The result is the natural loss of a good deal of information in the captured images. Without a proper paradigm to explain the mystery of human vision and perception, the recovery of lost information (reconstruction of the world) from 2D images represents a difficult hurdle for machine vision. However, despite this limitation, computer vision has progressed, riding mainly on the remarkable advancement of decade old digital image processing techniques, using the science and methods contributed by other disciplines such as optics, neurobiology, psychology, physics, mathematics, electronics, computer science, artificial intelligence and others.

Computer vision techniques and digital image processing methods in [1] both draw the proverbial water from the same pool, which is the digital image, and therefore necessarily overlap. Image processing takes a digital image and subjects it to processes, such as noise reduction, detail enhancement, or filtering, for producing another desired image as the result. For example, the blurred image of a car registration plate might be enhance by imaging techniques to produce a clear photo of the same so the police might identify the owner of the car. On the other hand, computer vision takes a digital image and subjects it to the same digital imaging techniques but for the purpose of analysing and understanding what the image depicts. For example, the image of a building can be fed to a computer and thereafter be identified by the computer as a residential house, a stadium, high-rise office tower, shopping mall, or a farm barn.

Russell and Norvig identified three broad approaches used in computer vision to distil useful information from the raw data provided by images. The first is the feature extraction approach, which focuses on simple computations applied directly to digital images to measure some useable characteristic, such as size. This relies on generally known image processing algorithms for noise reduction, filtering, object detection, edge detection, texture analysis, computation of optical flow, and segmentation, which techniques are commonly used to pre-process images for subsequent image analysis. This is also consider an “uninformed” approach.

The second is the recognition approach, where the focus is on distinguishing and labelling objects based on knowledge of characteristics that sets of similar objects have in common, such as shape or appearance or patterns of elements, sufficient to form classes. A classifier has to "learn" the patterns by being fed a training set of objects, their classes, achieving the goal of minimizing mistakes, and maximizing successes through a systematic process of improvement. There are many techniques in artificial intelligence that can be used for object or pattern recognition, including statistical pattern recognition, neural nets, genetic algorithms and fuzzy systems.

The third is the reconstruction approach, where the focus is on building a geometric model of the world suggested by the image or images and which is used as a basis for action. This corresponds to the stage of image understanding, which represents the highest and most complex level of computer vision processing. Here the emphasis is on enabling the computer vision system to construct internal models based on the data supplied by the images and to discard or update these internal models as they are verified against the real world or some other criteria. If the internal model is consistent with the real world, then image understanding takes place. Thus, image understanding requires the construction, manipulation and control of models and now relies heavily upon the science and technology of artificial intelligence.

2.2 OpenCV in Image Processing

OpenCv in [3] is a widely used tool in computer vision. It is a computer vision library for real-time applications, written in C and C++, which works with the Windows, Linux and Mac platforms. It is freely available as open source software from http://sourceforge.net/projects/opencvlibrary/.

OpenCv was start by Gary Bradsky at Intel in 1999 to encourage computer vision research and commercial applications and, side-by-side with these, promote the use of ever-faster processors from Intel. OpenCV contains optimised code for a basic computer vision infrastructure so developers do not have to re-invent the proverbial wheel. Bradsky and Kaehler in [5] provide the basic tutorial documentation. According to its website, OpenCV has been downloaded more than two million times and has a user group of more than 40,000 members. This attests to its popularity.A digital image is generally understood as a discrete number of light intensities captured by a device such as a camera and organized into a two-dimensional matrix of picture elements or pixels, each of which may be represented by number and all of which may be stored in a particular file format (such as jpg or gif). OpenCV goes beyond representing an image as an array of pixels. It represents an image as a data structure called an IplImage that makes immediately accessible useful image data or fields, such as:

- width – an integer showing the width of the image in pixels
- height – an integer showing the height of the image in pixels
- imageData – a pointer to an array of pixel values
- nChannels – an integer showing the number of colours per pixel
- depth – an integer showing the number of bits per pixel
- widthStep – an integer showing the number of bytes per image row
- geSize – an integer showing the size of in bytes
- roi – a pointer to a structure that defines a region of interest within the image.

OpenCV has a module containing basic image processing and computer vision algorithms. These include:

- smoothing (blurring) functions to reduce noise,
- dilation and erosion functions for isolation of individual elements,
- flood fill functions to isolate certain portions of the image for further processing,
- filter functions, including Sobel, Laplace and Canny for edge detection,
- Hough transform functions for finding lines and circles,
- Affine transform functions to stretch, shrink, warp and rotate images,
- Integral image function for summing sub regions (computing Haar wavelets),
- Histogram equalization function for uniform distribution of intensity values,
- Contour functions to connect edges into curves,
- Bounding boxes, circles and ellipses,
- Moments functions to compute Hu's moment invariants,

- Optical flow functions (Lucas-Kanade method),
- Motion tracking functions (Kalman filters), and
- Face detection/ Haar classifier.

OpenCV also has an ML (machine learning) module containing well-known statistical classifiers and clustering tools. These include:

- Normal/ naïve Bayes classifier,
- Decision trees classifier,
- Boosting group of classifiers,
- Neural networks algorithm, and
- Support vector machine classifier.

2.3 Pattern Recognition and Classifiers

In computer, vision a physical object maps to a particular segmented region in the image from which object descriptors or features may be derive. A feature is any characteristic of an image, or any region within it, that can be measure. Objects with common features may be group into classes, where the combination of features may be considered a pattern. Object recognition in [2] may be understood to be the assignment of classes to objects based on their respective patterns. The program that does this assignment is called a classifier.

The general steps in pattern recognition may be summarized in Figure below:

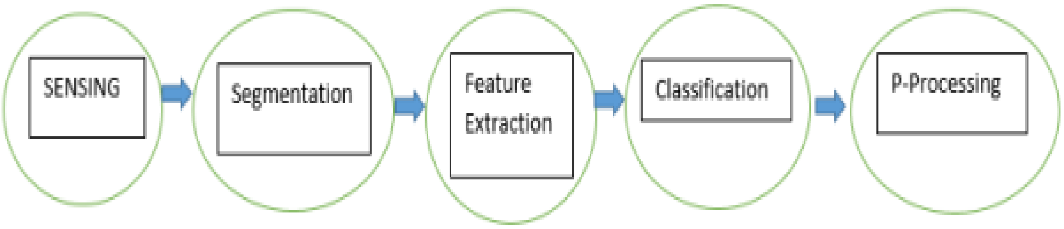


Figure 2.1: General Pattern Recognition Steps

The most important step is the design of the formal descriptors because choices have to be made on which characteristics, quantitative or qualitative, would best suit the target object and in turn determines the success of the classifier.

In statistical pattern in [5] recognition, quantitative descriptions called features are used. The set of features constitutes the pattern vector or feature vector, and the set of all possible patterns for the object form the pattern space X (also known as feature space). Quantitatively, similar objects in each class will be located near each other in the feature space forming clusters, which may ideally be separated from dissimilar objects by lines or curves called discrimination functions. Determining the most suitable discrimination function or discriminant to use is part of classifier design.

A statistical classifier accepts n features as inputs and gives 1 output, which is the classification or decision about the class of the object. The relationship between the inputs and the output is a decision rule, which is a function that puts in one space or subset those feature vectors that are associated with a particular output. The decision rule is based on the particular discrimination function used for separating the subsets from each other.

The ability of a classifier to classify objects based on its decision rule may be understood as classifier learning discussed in [3] , and the set of the feature vectors (objects) inputs and corresponding outputs of classifications (both positive and negative results) is called the training set. It is expected that a well-designed classifier should get 100% correct answers on its training set. A large training set is generally desirable to optimize the training of the classifier, so that it may be tested on objects it has not encountered before, which constitutes its test set. If the classifier does not perform well on the test set, modifications to the design of the recognition system may be needed.

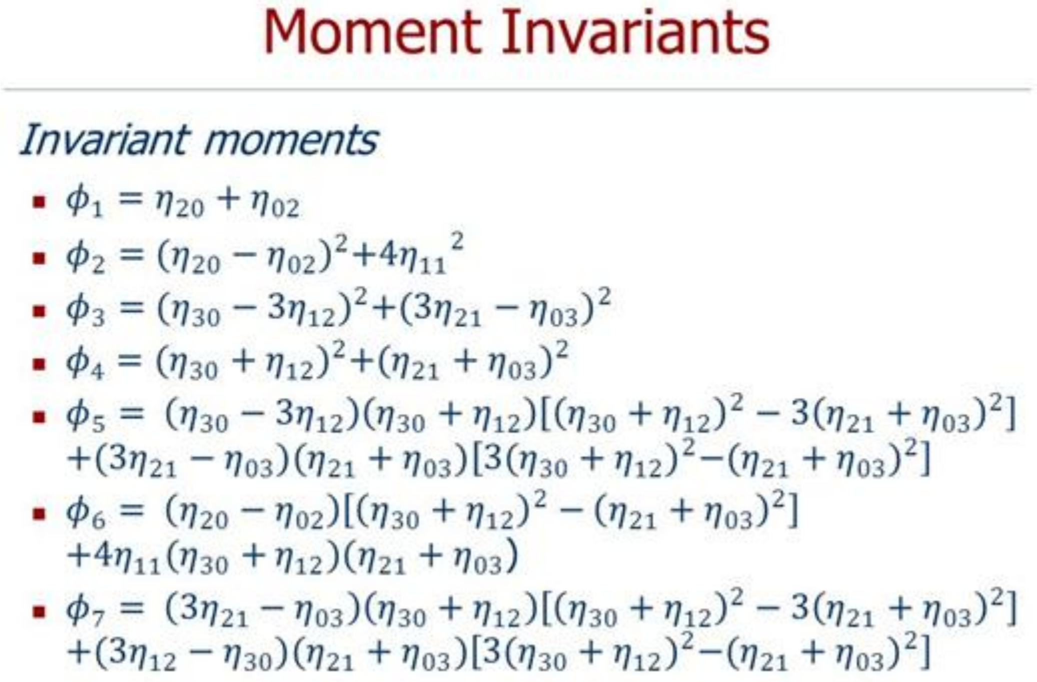
2.4 Moment Invariants in Image Processing

As mentioned in [3] , feature extraction is one approach used in computer vision. According to A.L.C. Barczak, feature extraction refers to the process of distilling a limited number of features that would be sufficient to describe a large set of data, such as the pixels in a digital image . The idea is to use the features as a unique representation of the image.

Since a digital image is a two-dimensional matrix of pixels values, region-based object descriptions are affected by geometric transformations, such as scaling, translation, and rotation. For example, the numerical features describing the shape of a 2D object would change if the shape of the same object changes as seen from a different angle or perspective. However, to be useful in computer vision applications, object descriptions must be able to identify the same object irrespective of its position, orientation, or distortion.

One of the most popular quantitative object descriptors are moments. Hu first formulated the concept of statistical characteristics or moments that would be indifferent to geometric transformations in 1962. Moments are polynomials of increasing order that describe the shape of a statistical distribution . Its exponent indicates the order of a moment. The geometric moments of different orders represent different spatial characteristics of the image intensity distribution. A set of moments can thus form a global shape descriptor of an image.

Hu proposed that the following seven functions (called 2D moment invariants) were invariant to translation, scale variation, and rotation of an image in 2.2.Since they are invariant to geometric transformations, a set of moment invariants computed for an image may be considered as a feature vector. A set of feature vectors might constitute a class for object detection and recognition. The feature vectors of a class of reference images can be compared with the feature vectors of the image of an unknown object, and if their feature vectors do not match, then they may be considered as different objects. The usefulness of moment invariants as image shape descriptors in pattern recognition and object identification is well established. A code fragment implementing an approximation of the first of Hu's moment invariants is presented in the next section. OpenCV has built-in functions for the calculation of moments: cvMoments(), cvGetCentralMoment(), cvGetNormalizedCentralMoment() and cvGetHuMoments().



Moment Invariants

Invariant moments

- $\phi_1 = \eta_{20} + \eta_{02}$
- $\phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}{}^2$
- $\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$
- $\phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$
- $\phi_5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2]$ $+ (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$
- $\phi_6 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$ $+ 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})$
- $\phi_7 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2]$ $+ (3\eta_{12} - \eta_{30})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$

Figure 2.2: Moment Invariants

2.5 Otsu Thresholding Algorithm For Pattern Recognition

In computer vision and image processing, **Otsu's method in [2]**, named after Nobuyuki Otsu & it is used to perform automatic image thresholding. In the simplest form, the algorithm returns a single intensity threshold that separate pixels into two classes, foreground and background. This threshold is determined by minimizing intra-class intensity variance, or equivalently, by maximizing inter-class variance. Otsu's method is a one-dimensional discrete analog of Fisher's Discriminant Analysis, is related to Jenks optimization method, and is equivalent to a globally optimal k-means performed on the intensity histogram. The extension to multi-level thresholding was described in the original paper and computationally efficient implementations have since been proposed.

2.5.1 Otsu Method

Otsu’s thresholding method in [2] corresponds to the linear discriminant criteria that assumes that the image consists of only object (foreground) and background, and the heterogeneity and diversity of the background is ignored. Otsu set the threshold to try to minimize the overlapping of the class distributions. Given this definition, the Otsu’s method segments the image into two light and dark regions T0 and T1, where region T0 is a set of intensity level from 0 to t or in set notation T0 = {0, 1,...,t} and region T1 = {t,t + 1,...,l−1,l} where t is the threshold value, l is the image maximum gray level (for instance 256). T0 and T1 can be assigned to object and background or vice versa (object not necessarily always occupies the light region). Otsu's thresholding method scans all the possible thresholding values and calculates the minimum value for the pixel levels each side of the threshold. The goal is to find the threshold value with the minimum entropy for sum of foreground and background. Otsu's method determines the threshold value based on the statistical information of the image where for a chosen threshold value t the variance of clusters T0 and T1 can be computed. The optimal threshold value is calculated by minimizing the sum of the weighted group variances, where the weights are the probability of the respective groups. Given: p (i) as the histogram probabilities of the observed gray value i=1,...,l

$$P\ (i) = \text{number}\ \{(r, c)\ |\text{image}(r, c) = i\}\ (R, C)$$

Where r, c is index for row and column of the image, respectively, R and C is the number of rows and columns of the image, respectively. $wb(t)$, $\mu b(t)$, and $\sigma 2\ b(t)$ as the weight, mean, and variance of class T0 with intensity value from 0 to t, respectively. $wf(t)$, $\mu f(t)$, and $\sigma 2\ f(t)$ as the weight, mean, and variance of class T1 with intensity value from t+1 to l, respectively. $\sigma 2\ w$ as the weighed sum of group variances. The best threshold value t* is the value with the minimum within class variance. The within class variance defines as following:

$$\sigma 2\ w = wb(t) * \sigma 2\ b(t) + wf(t) * \sigma 2\ f(t)$$

where $wb(t) = \sum t\ i=1\ P(i)$, $wf(t) = \sum l\ i=t+1\ P(i)$, $\mu b(t) = \sum t\ i=1\ i * P(i)\ wb(t)$; $\mu f(t) = \sum l\ i=t+1\ i * P(i)\ wf(t)$; $\sigma 2\ b(t) = \sum t\ i=1 (i - \mu b(t)) 2 * P(i)\ wb(t)$ and $\sigma 2\ f(t) = \sum l\ i=t+1 (i - \mu f(t)) 2 * P(i)\ wf(t)$.

CHAPTER 3

SYSTEM REQUIREMENTS SPECIFICATION

A System Requirement Specification (SRS) is an organization's understanding of a customer or potential client's system requirements and dependencies at a particular point prior to any actual design or development work. The information gathered during the analysis is translated into a document that defines a set of requirements. It gives the brief description of the services that the system should provide and the constraints under which, the system should operate. Generally, SRS is a document that completely describes what the proposed software should do without describing how the software will do it. A two-way insurance policy assures that both the client and the organization understand the other's requirements from that perspective at a given point in time.

SRS document itself states in precise and explicit language those functions and capabilities a software system (i.e., a software application, an ecommerce website and so on) must provide, as well as states any required constraints by which the system must abide. SRS also functions as a blueprint for completing a project with as little cost growth as possible. SRS is often referred to as the "parent" document because all subsequent project management documents, such as design specifications, statements of work, software architecture specifications, testing and validation plans, and documentation plans, are related to it.

Requirement is a condition or capability to which the system must conform. Requirement Management is a systematic approach towards eliciting, organizing and documenting the requirements of the system clearly along with the applicable attributes. The elusive difficulties of requirements are not always obvious and can come from any number of sources.

3.1 General Description

In this section of the presented thesis, the introduction of software product under consideration has been presented. It presents the basic characteristics and factors influencing the software product or system model and its requirements.

3.1.1 Product Perspective

In this project or research work, we have proposed a highly robust and efficient media control hand gesture detection system and gesture recognition based application command generation scheme. The proposed system has been emphasized on developing an efficient scheme that can accomplish hand gesture recognition without introducing any training related overheads. The proposed system has to take into consideration of geometrical shape of human hand and based on defined thresholds and real time parametric variation, the segmentation for human shape is accomplished. Based on retrieved specific shape, certain application-oriented commands have to be generated. The predominant uniqueness of the proposed scheme is that it does not employ any kind of prior training and it is functional in real time without having any databases or training datasets. Unlike tradition approaches of images, datasets based recognition system; this approach achieves hand gesture recognition in real time, and responds correspondingly. This developed mechanism neither introduce any computational complexity nor does it cause any user interferences to achieve tracing of human gesture.

3.1.2 User Characteristics

The user should have at least a basic knowledge of windows and web browsers, such as install software like OpenCV, Python etc. and executing a program, and the ability to follow on screen instructions. The user will not need any technical expertise in order to use this program.

3.2 Functional Requirements

- The camera used will be able to capture user images from the video sequences.
- The software will be able to produce multiple frames and display the image in the RGB colour space.

- The software will be able to display the converted RGB image in a new window and convert it into grey image.
- The software will be able to detect the contours of the detected skin regions.
- The software, which act as an intermediate in passing these, processed image in order to control the media player.

3.3 Non-functional Requirements

- **Usability:** The user is facilitated with the control section for the entire process in which they can arrange the position of hand at the centre of ROI under consideration, the variation of palm position and respective command generation etc. can be effectively facilitated by mean of user interface. The implementation and calibration of camera and its resolution can also be done as per quality and preciseness requirement.

  The frame size, flow rate and its command variation with respect to threshold developed and colour component of hand colour, can be easily calibrated by means of certain defined thresholds.
- **Security and support:** Application will be permissible to be used only in secure network so there is less feasibility of insecurity over the functionality of the application. On the other hand, the system functions in a real time application scenario, therefore the camera, colour and platform compatibility is must in this case. IN case of command transfer using certain connected devices or wireless communication, the proper port assignment would also be a predominant factor to be consider.
- **Maintainability:** The installation and operation manual of the project will be provided to the user.
- **Extensibility:** The project work is also open for any future modification and hence the work could be define as the one of the extensible work.

3.4 External Interface Requirements

An interface description for short is a specification used for describing a software component's interface. IDLs are commonly used in remote procedure call software. In these issues, the machines on moreover last part of the "link" might be utilizing Dissimilar Interface Description recommends a bridge between the two diverse systems. These descriptions are classified into following types:

- **User Interface:** The external or operating user is an individual who is interested to introduce a novel Algorithm for shape based hand gesture recognition in real time application scenario. The user interface would be like axis presenting real time movement of human hand and its relative position with respect to defined centroid or morphological thresholds.
- **Restoration with Text Removal Software Interface:** The Operating Systems can be any version of Windows, Linux, UNIX or Mac.
- **Hardware Interface:** In the execution of this project, the hardware interface used is a normal 32/64 bit operating system supported along with better integration with network interface card for better communication with other workstations. For better and precise outcome, a high definition camera with calibrated functioning with defined RGB or YBR colour format is must. Since the proposed system functions in real time application, therefore the camera quality and its colour accuracy would be important. In the proposed system, the background also plays a vital role, therefore the background segmentation or calibration with well-defined frame rate or resolution would be must. Such cautions would ensure optimal recognition and tracing of hand gesture.

CHAPTER 4

SYSTEM ANALYSIS

Analysis is the process of finding the best solution to the problem. System analysis is the process by which we learn about the existing problems, define objects and requirements and evaluates the solutions. It is the way of thinking about the organization and the problem it involves, a set of technologies that helps in solving these problems. Feasibility study plays an important role in system analysis, which gives the target for design and development.

4.1 Feasibility Study

Gesture Recognition systems are feasible when provided with unlimited resource and infinite time. But unfortunately this condition does not prevail in practical world. So it is both necessary and prudent to evaluate the feasibility of the system at the earliest possible time. Months or years of effort, thousands of rupees and untold professional embarrassment can be averted if an ill-conceived system is recognized early in the definition phase. Feasibility & risk analysis are related in many ways. If project risk is great, the feasibility of producing quality software is reduced. In this case three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

4.1.1 Economical Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund ,used to develop the Digital Image Processing System for research and development of the system is limited. The expenditures must be justified. Thus, the developed system i.e. Hand Gesture Detection System ,as well within the budget and this was achieved because most of the technologies used are freely available.

4.1.2 Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. OpenCV (Open Source Computer Vision Library) is a library which mainly focuses at real-time computer vision. It is free for both academic and commercial use. It has C++, C, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. OpenCV was designed for computational efficiency and with a strong focus on real-time applications. The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. It provides basic data structures for image processing with efficient optimizations

4.1.3 Social Feasibility

The aspect of study in Hand Gesture Detection System is to check the level of acceptance of the gesture of the user by the system . This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods to recognize the number of fingers raised in front of the camera that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcome, as he is the final user of the system.

4.2 Analysis

4.2.1 Performance Analysis

For the complete functionality of the project work, the project is run with the help of healthy networking environment. Performance analysis is done to find out whether the proposed system. It is essential that the process of performance analysis and definition must be conduct in parallel.

4.2.2 Technical Analysis

Gesture Detection System is only beneficial only if it can be turn into information systems that will meet the organization’s technical requirement. Simply stated this test of feasibility asks whether the system will work or not when developed & installed, whether there are any major barriers to implementation. Regarding all these issues in technical analysis there are several points to focus on:

Changes to bring in the system: All changes done to recognize the hand gesture should be in positive direction, there would be increased level of efficiency and better customer service.

Required skills: Platforms such Spyder(in Anaconda) & libraries such openCV,PyautoGUI used in this project are widely used. Therefore, the skilled work force is readily available in the industry.

Acceptability: The structure of the system is kept feasible enough so that there should not be any problem from the user’s point of view.

4.2.3 Economical Analysis

Economic analysis in Gesture Processing and detecting system is done to perform & to evaluate the development cost weighed against the ultimate income or benefits derived from the developed system. For running this system, we need not have any routers, which are highly economical. Therefore, the system is economically feasible enough.

CHAPTER 5

SYSTEM DESIGN

Design is a meaningful engineering representation of something that is to be built. It is the most crucial phase in the developments of a system. Software design is a process through which the requirements are translated into a representation of software. Design is a place where design is fostered in software Engineering. Based on the user requirements and the detailed analysis of the existing system, the new system must be designed. This is the phase of system designing. Design is the perfect way to accurately translate a customer's requirement in the finished software product. Design creates a representation or model, provides details about software data structure, architecture, interfaces and components that are necessary to implement a system. The logical system design arrived at because of systems analysis is converted into physical system design.

5.1 System development methodology

System development method is a process through which a product will get completed or a product gets rid from any problem. Software development process is described as a number of phases, procedures and steps that gives the complete software. It follows series of steps, which are used for product progress. The development method followed in this project is waterfall model.

5.1.1 Model phases

The waterfall model is a sequential software development process, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of Requirement initiation, Analysis, Design, Implementation, Testing and maintenance.

Requirement Analysis: This phase is concerned about collection of requirement of the system. This process involves generating document and requirement review.

System Design: Keeping the requirements in mind the system specifications are translate in to a software representation. In this phase, the designer emphasizes on-algorithm, data structure, software architecture etc.

Coding: In this phase, programmer starts his coding in order to give a full sketch of product. In other words, system specifications are only convert in to machine-readable compute code.

Implementation: The implementation phase involves the actual coding or programming of the software. The output of this phase is typically the library, executables, user manuals and additional software documentation

Testing: In this phase, all programs (models) are integrated and tested to ensure that the complete system meets the software requirements. The testing is concerned with verification and validation.

Maintenance: The maintenance phase is the longest phase in which the software is updated to fulfil the changing customer need, adapt to accommodate change in the external environment, correct errors and oversights previously undetected in the testing phase, enhance the efficiency of the software.

5.1.2 Advantages of the Waterfall Model

- Clear project objectives.
- Stable project requirements.
- Progress of system is measurable.
- Strict sign-off requirements.
- Helps you to be perfect.
- Logic of software development is clearly understood.
- Production of a formal specification.
- Better resource allocation.

- Improves quality. The emphasis on requirements and design before writing a single line of code ensures minimal wastage of time and effort and reduces the risk of schedule slippage.
- Less human resources required as once one phase is finished those people can start working on to the next phase.

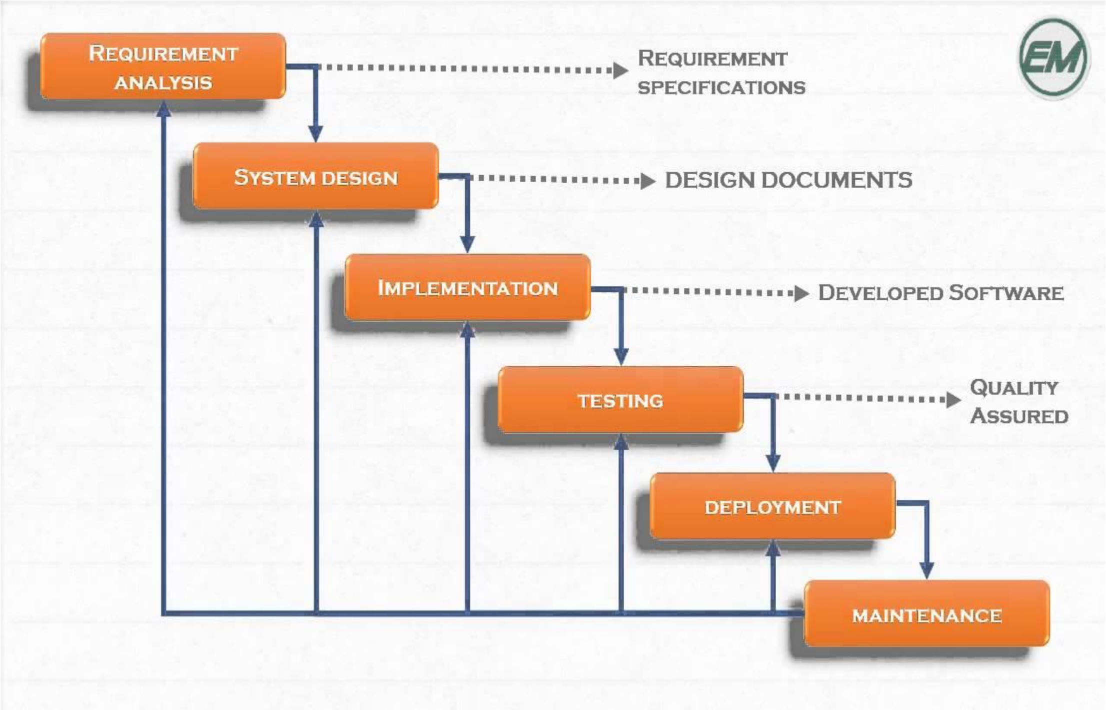
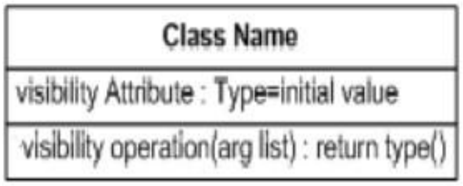
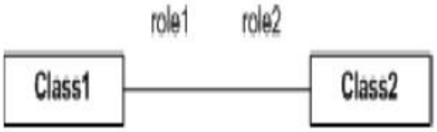
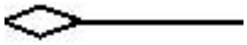
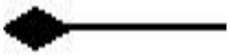


Figure 5.1: Waterfall Model

5.2 Design Using UML

Designing UML diagram specifies, how the process within the system communicates along with how the objects with in the process collaborate using both static as well as dynamic UML diagrams since in this ever-changing world of Object Oriented application development, it has been getting harder and harder to develop and manage high quality applications in reasonable amount of time. Because of this challenge and the need for a universal object modelling language every one could use, the Unified Modelling Language (UML) is the Information industries version of blue print. It is a method for describing the systems architecture in detail. Easier to build or maintains system, and to ensure that the system will hold up to the requirement changes

Sl. No	Symbol Name	Symbol	Description
1	Class	 Class Name visibility Attribute : Type=initial value visibility operation(arg list) : return type()	Classes represent a collection of similar entities grouped together.
2	Association		Association represents a static relation between classes
3	Aggregation		Aggregation is a form of association. It aggregates several classes into a single class.
4	Composition		Composition is a special type of aggregation that denotes a strong ownershipbetween classes.

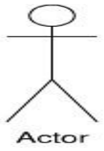
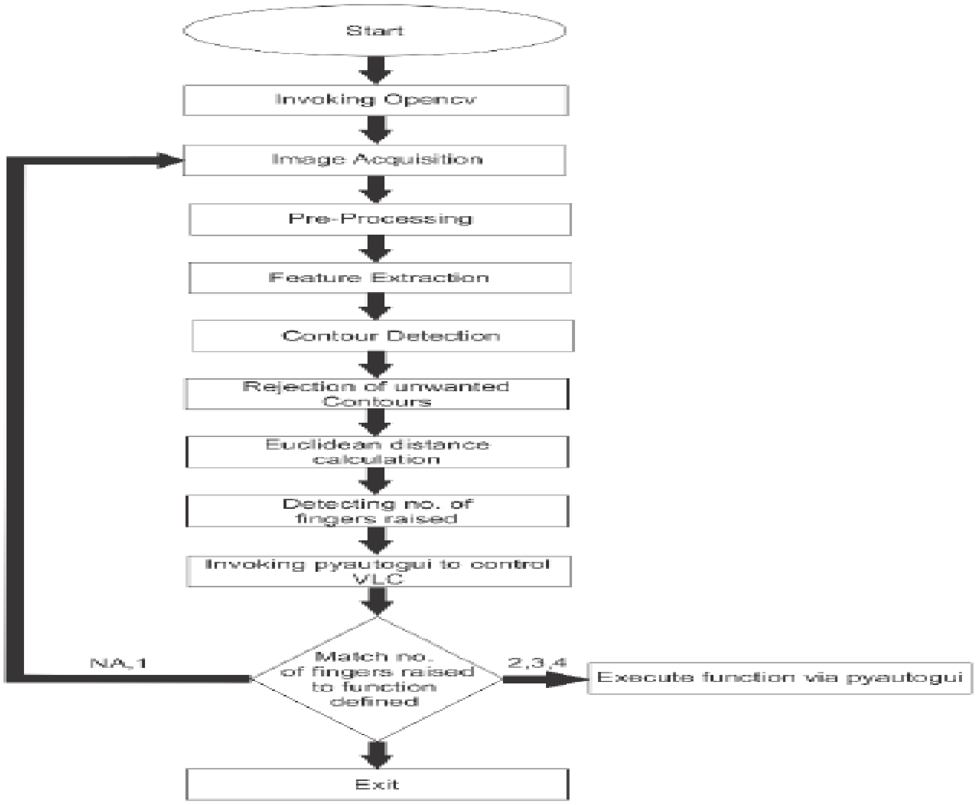
5	Actor		Actor is the user of the system that reacts with the system.

Table 5.1: Symbols used in UML

5.3 Data Flow Diagram

The Given DFD in figure 5.2 shows the flowchart of the implementation of the system which is explain in chapter 6 i.e. Proposed System.



Start
Invoking Opencv
Image Acquisition
Pre-Processing
Feature Extraction
Contour Detection
Rejection of unwanted Contours
Euclidean distance calculation
Detecting no. of fingers raised
Invoking pyautogui to control VLC
Match no. of fingers raised to function defined
NA,1
2,3,4
Execute function via pyautogui
Exit

Figure 5.2: Data Flow Diagram

The data flow diagram essentially shows how the data control flows from one module to another. Unless the input filenames are correctly given the program cannot proceed to the next module. Once the user gives, the correct input filenames parsing is done individually for each file. The required information is taken in parsing and an adjacency matrix is generated for that. From the adjacency matrix, a lookup table is generated giving paths for blocks. In addition, the final sequence is computed with the lookup table and the final required code is generated in an output file. In case of multiple file inputs, the code for each is generated and combined together.

5.4 COMPONENT DIAGRAM

In the Unified Modelling Language, a given component diagram in figure 5.3 depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems.

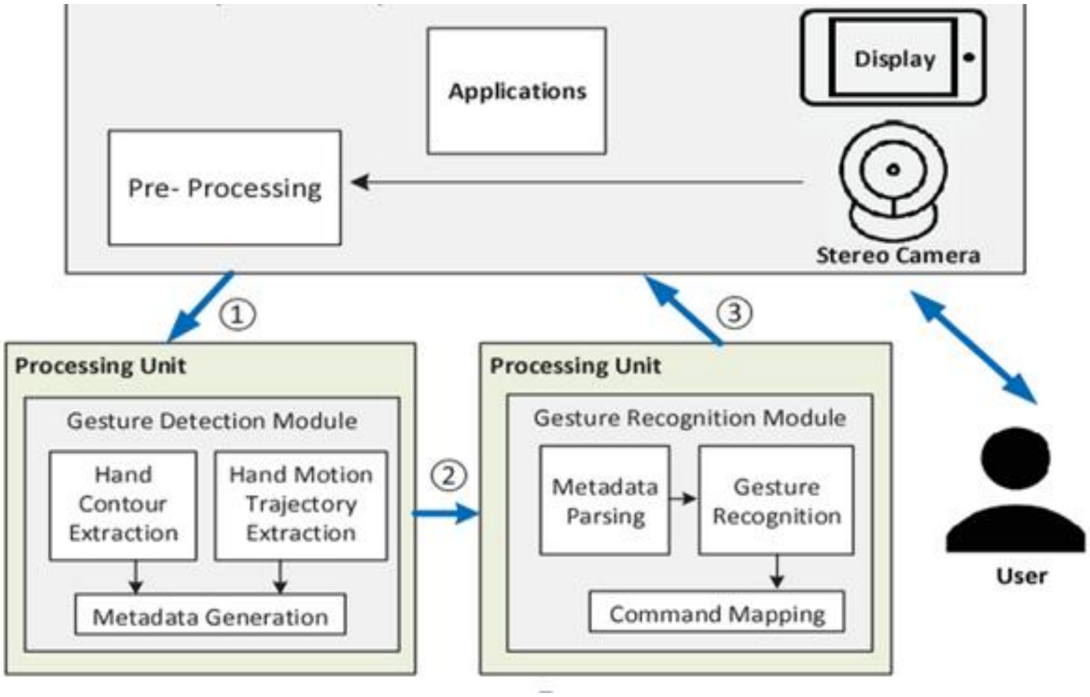


Figure 5.3: Component Diagram

The component diagram for the gesture detection System include the various unit for input and output operation. For our designed we have mainly two process in which one is to capture the image through camera which is done by invoking openCV and other is pre-processing done by the system. The processing includes two Unit which is used to process the image capture by the camera, Firstly the preprocessing unit detects the metadata ig the image and its trajectory that is the orientation in which the hand fingers was raised ,then its sent for further pre-processing.In Further pre-processing system used the algorithm for extracting the feature to recognize the fingers raised The feature are extracted using the metadata and information in previous pre-processing steps.It is interesting to note that all the sequence of activities that are taking place are via this module itself, i.e. the parsing and the process of computing the final sequence. The parsing redirects across the other modules until the final code is generated.

5.5 Use case Diagram

A use case defines a goal-oriented set of interactions between external entities and the system under consideration. The external entities, which interact with the system, are its actors such as user and sytems for our Application. A set of use cases describe the complete functionality of the system at a particular level of detail and the use case diagram can graphically denote it.

The shown Use case diagram in figure 5.4 explain the different entities involved in making interaction with the System. As this System is based on Human Computer Interaction so it basically include the user, computer and the medium to connect both digitally that is web camera which is connected to System. As the execution of the program starts, it firstly invoke the web camera to take RGB image of the hand. Then the image is segmented and filtered to reduce the noise in the image .After the removal of the noise the hand gestures are detected i.e the number of fingers raised are preprocessed, on the basis of feature are extracted. After Feature extraction is done, By using conditionals statements in the program the feature are matched .As the features matches ,it automatically controls the media player and gives us the required results.

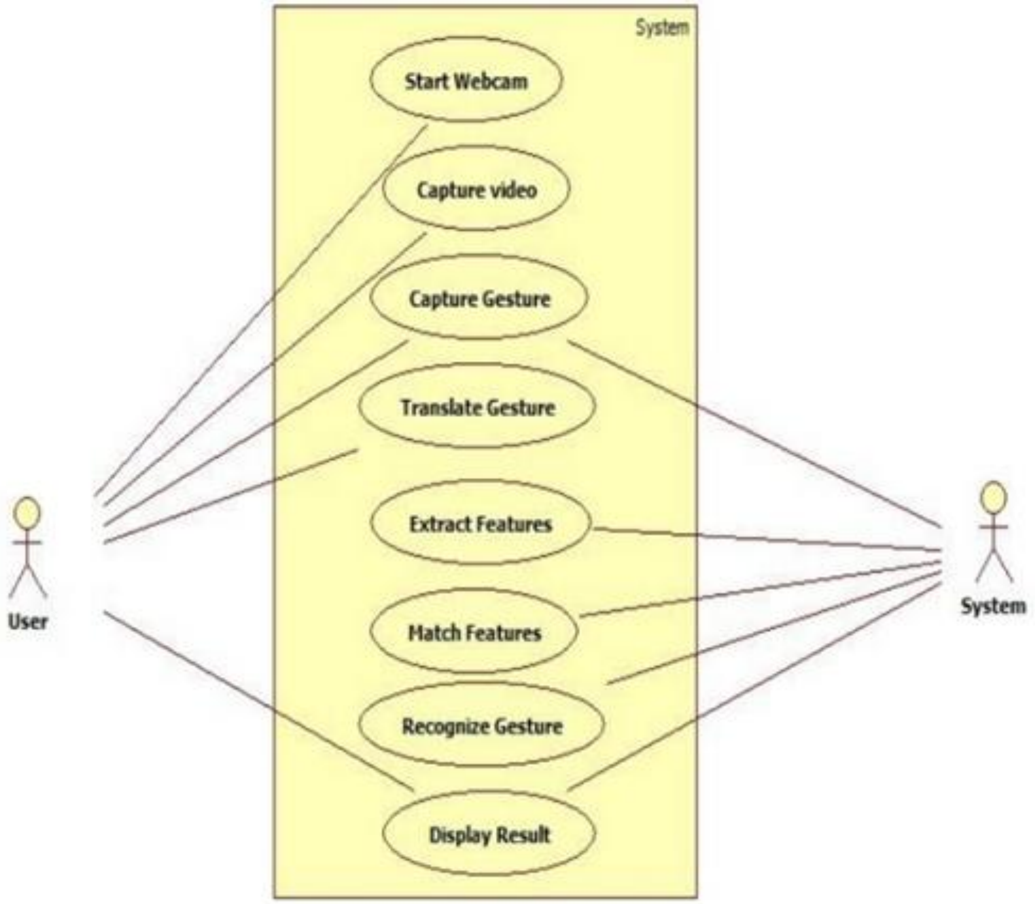


Figure 5.4: Use Case Diagram

5.6 Activity Diagram

An activity diagram shows the sequence of steps that make up a complex process. An activity is shown as a round box containing the name of the operation. An outgoing solid arrow attached to the end of the activity symbol indicates a transition triggered by the completion.

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams are intended to model both computational and organisational processes (i.e. workflows). Activity diagrams show the overall flow of control. Activity diagrams constructed from a limited number of shapes, connected with arrows. The most important shape types:

- rounded rectangles represent actions;
- diamonds represent decisions;
- bars represent the start (split) or end (join) of concurrent activities;
- a black circle represents the start (initial state) of the workflow;
- An encircled black circle represents the end (final state).

The basic purposes of activity diagrams are similar to other four diagrams. It captures the dynamic behaviour of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another.

Activity is a particular operation of the system. Activity diagrams are not only use for visualizing dynamic nature of a system but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in activity diagram is the message part.

Recognition of hand Gesture includes various activities to be performed. As shown in the figure 5.5 the first activity is to start the camera to capture the image. This activity automatically invoke the camera using openCV library in python as the execution of the program starts.Then on the basis of the capture image , the gesture information is extracted. This information is used to extract the features such contours , convex hull and the defects point. On the basis of which the number of fingers in front of the camera is Recognized .As the Finger information is extracted ,the application automatically perform the action such as play, pause, seek forward, seek backward etc on the basis of number of fingers raise, on the VLC Media Player .

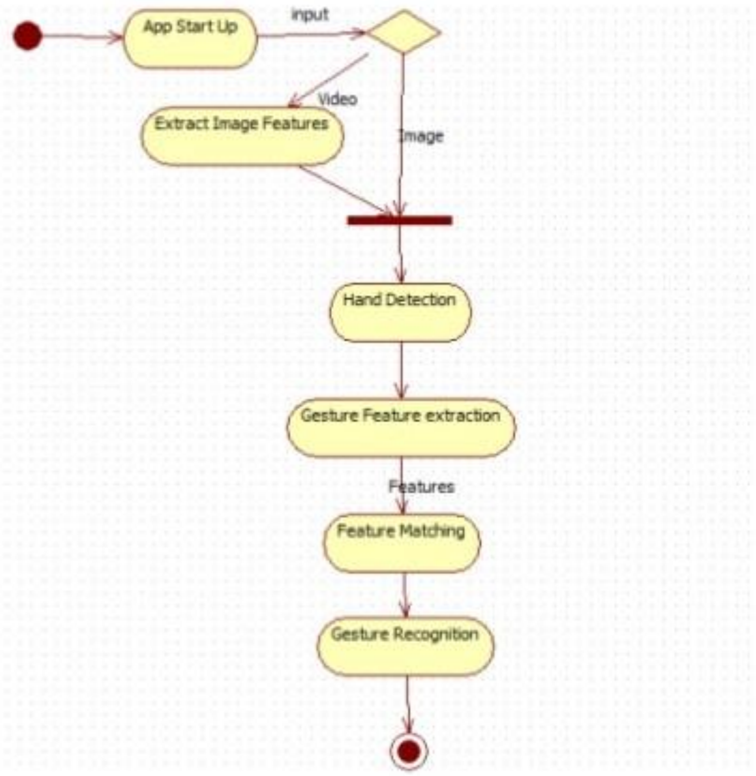


Figure 5.5: Activity Diagram

5.7 Sequence Diagram

Sequence diagram are an easy and intuitive way of describing the behaviour of a system by viewing the interaction between the system and the environment. A sequence diagram shows an interaction arranged in a time sequence. A sequence diagram has two dimensions: vertical dimension represents time; the horizontal dimension represents the objects existence during the interaction.

The shown Sequence diagram explain the flow of the program. As this System is based on Human Computer Interaction so it basically include the user, computer and the medium to connect both digitally that is web camera. As the execution of the program starts, it firstly invoke the web camera to take RGB image of the hand. Then the image is segmented and filtered to reduce the noise in the image .After the removal of the noise the hand gestures are detected i.e the number of fingers raised are preprocessed, on the basis of feature are extracted. After Feature extraction is done, By using conditionals statements in the program the feature are matched .As the features matches ,it automatically controls the media player and gives us the required results.

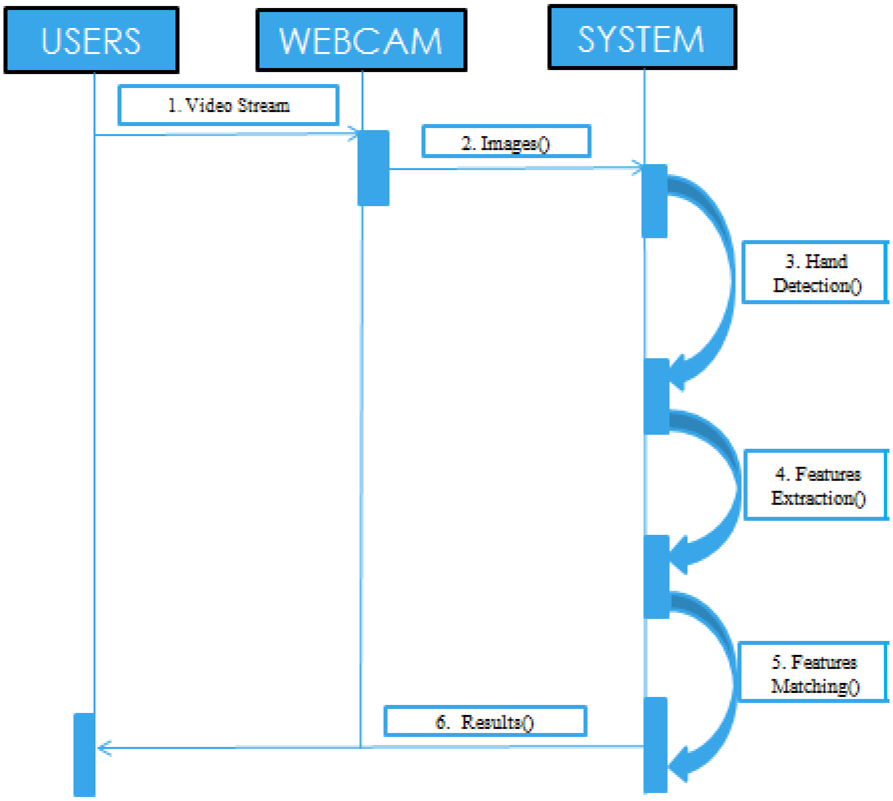


Figure 5.6: Sequence Diagram

CHAPTER 6

PROPOSED SYSTEM

6.1 Methodology

This project availed of several algorithms commonly used in computer vision. These include those used in colour segmentation, Morphological filtering, Extraction of Features, Contours, Convex Hull and controlling media player using pyautogui.

6.1.1 Data Obtaining

The initial move is to capture the image from camera and to define a region of Interest in the frame, it is important as the image can contain a lot of variables and these variables can result in unwanted results and the data that needs to be processed is reduced to a large extent. To capture the image a web-camera is used that continuously captures frames and is used to get the raw data for processing. The input picture we have here is uint8. The Procured image is RGB and must be process before i.e. pre-processed before the components are separated and acknowledgement is made.

6.1.2 Colour Segmentation Using Thresholding

Segmentation is the process of identifying regions within an image. Colour can be used to help in segmentation. In this project, the hand on the image was the region of interest. To isolate the image pixels of the hand from the background, the range of the HSV values for skin colour was determined for use as the threshold values. Segmentation could then proceed after the conversion of all pixels falling within those threshold values to white and those without to black.

The algorithm used for the colour segmentation using thresholding is shown below:

- Capture an image of the gesture from the camera.

- Determine the range of HSV values for skin colour for use as threshold values.
- Convert the image from RGB colour space to HSV colour space
- Convert all the pixels falling within the threshold values to white.
- Convert all other pixels to black.
- Save the segmented image in an image file.

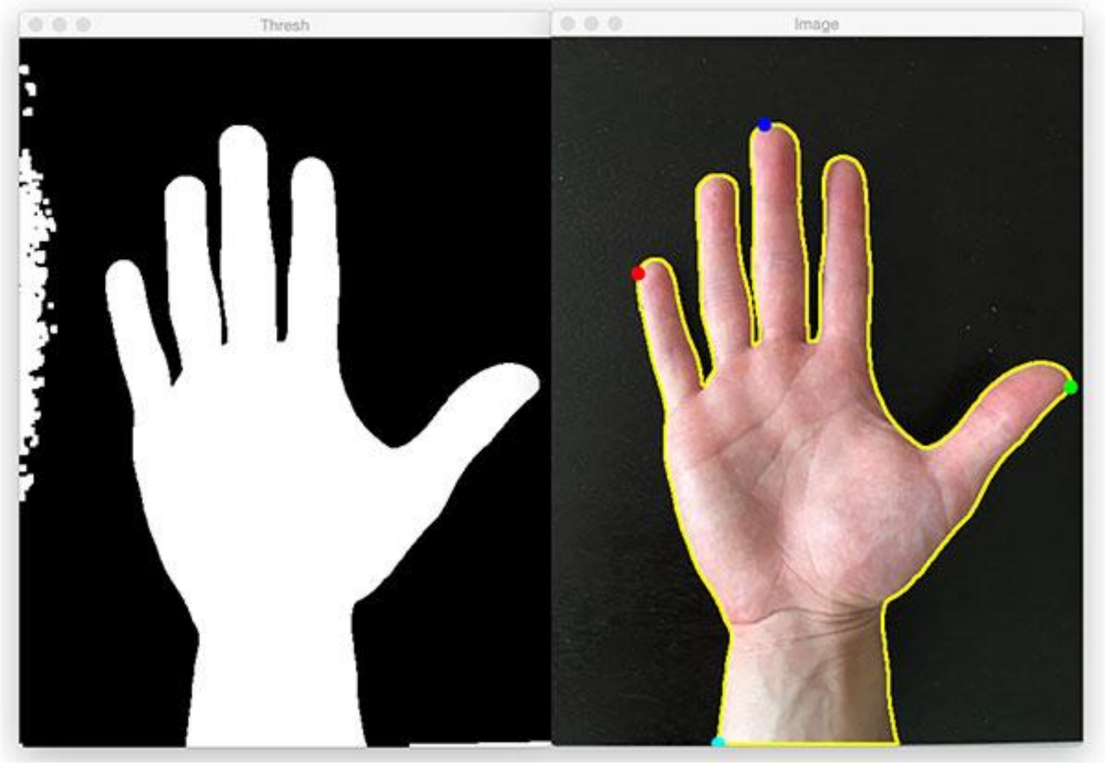


Figure 6.1: Segmentation

6.1.3 Morphological Filtering

Morphological image processing is a collection of non-linear operations related to the shape or morphology of features in an image. According to Wikipedia, morphological operations rely only on the relative ordering of pixel values, not on their numerical values, and therefore are especially suited to the processing of binary images. Morphological operations can also be applied to greyscale images such that their light transfer functions are unknown and therefore their absolute pixel values are of no or minor interest. After thresholding we have to make sure that there will be no noise is present in image, so we are using morphological filtering Techniques, These Techniques are divided in-

6.1.3.1 Dilation

Dilation is a process in which the binary image is expanded from its original shape. The way the binary image is expanded is determined by the structuring element.

This structuring element is smaller compare to the image itself, and normally the size used for the structuring element is 3 x 3.

The dilation process is similar to the convolution process, that is, the structuring element is reflected and shifted from left to right and from top to bottom, at each shift; the process will look for any overlapping similar pixels between the structuring element and that of the binary image.

If there exists an overlapping then the pixels under the center position of the structuring element will be turned to 1 or black.

Let us define X as the reference image and B as the structuring clement. The dilation operation is defined by equation,

$$X \oplus B = \{z | [(B^\wedge)Z \cap X] \in X\} X \oplus B = \{z | [(B^\wedge)Z \cap X] \in X\}$$

Where B is the image, B rotated about the origin. Equation states that when the image X is dilated by the structuring element B, the outcome element z would be that there would be at least one element in B that intersects with an element in X.

If this is the case, the position where the structuring element is being centered on the image will be 'ON'. This process is illustrated in Fig. a. The black square represents I and the white square represents 0.

Initially, the center of the structuring element is aligned at position •. At this point, there is no overlapping between the black squares of B and the black squares of X; hence at position • the square will remain white.

This structuring element will then be shifted towards right. At position **, we find that one of the black squares of B is overlapping or intersecting with the black square of X. Thus, at position • • the square will be changed to black. Similarly, the structuring element B is shift from left to right and from top to bottom on the image X to yield the dilated image as shown in Fig. 6.2.

The dilation is an expansion operator that enlarges binary objects. Dilation has many uses, but the major one is bridging gaps in an image, because B is expanding the features of X.

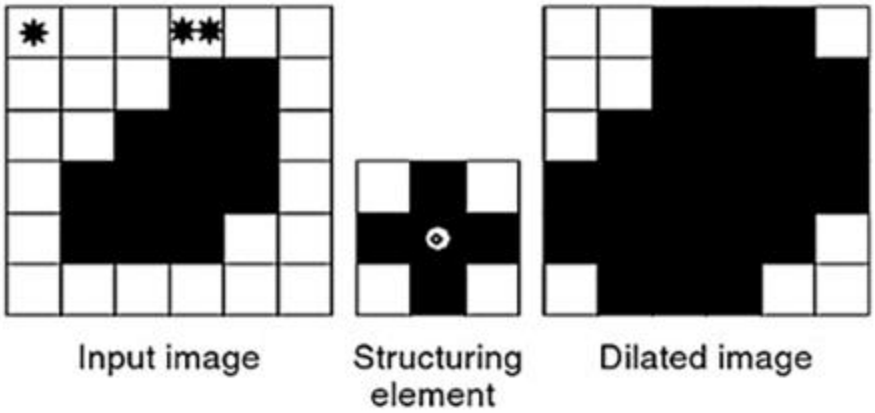


Figure 6.2: Dilation

6.1.3.2 Erosion

Erosion is the counter-process of dilation. If dilation enlarges, an image then erosion shrinks the image. The way the image is shrunk is determined by the structuring element. The structuring element is normally smaller than the image with a 3 x 3 size.

This will ensure faster computation time when compared to larger structuring-element size. Almost similar to the dilation process, the erosion process will move the structuring element from left to right and top to bottom.

At the center position, indicated by the center of the structuring element, the process will look for whether there is a complete overlap with the structuring element or not.If there is no complete overlapping then the center pixel indicated by the center of the structuring element will be set white or 0. Let us define X as the reference binary image and B as the structuring element. Erosion is define by the equation

$$X \ominus B = \{z|(B^\wedge)Z \in X\} X \ominus B = \{z|(B^\wedge)Z \in X\}$$

Equation states that the outcome element z is consider only when the structuring element is a subset or equal to the binary image X. This process is depicted in Fig. 6.3. Again, the white square indicates O and the black square indicates one.

The erosion process starts at position •. Here, there is no complete overlapping, and so the pixel at the position • will remain white.The structuring element is then shifted to the right and the same condition is observed. At position u, complete overlapping is not present; thus, the black square marked with • • will be turned to white.

The structuring element is then shifted further until its center reaches the position marked by •• •. Here, we see that the overlapping is complete, that is, all the black squares in the structuring element overlap with the black squares in the image.Hence, the center of the structuring element corresponding to the image will be black. Figure b shows the result after the structuring element has reached the last pixel. Erosion is a thinning operator that shrinks an image. By applying erosion to an image, narrow regions can be eliminated while wider ones are thinned.

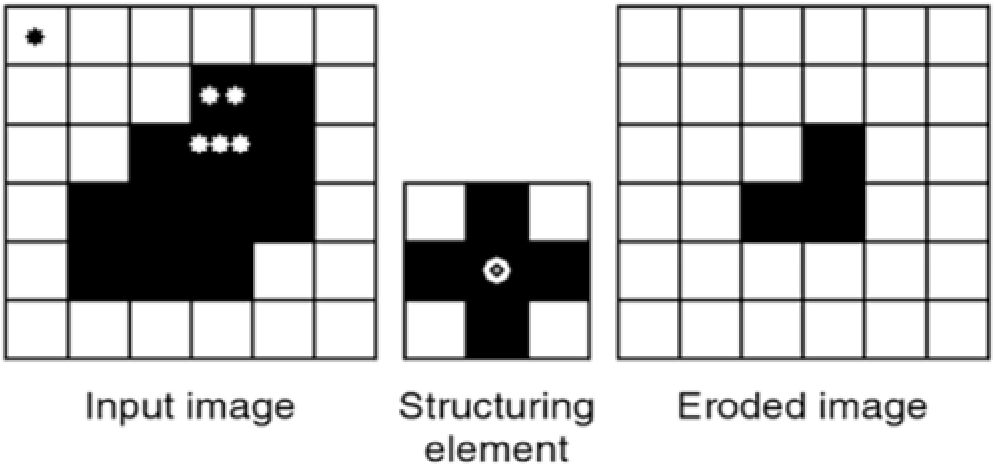


Figure 6.3: Erosion

If the division is not continuous, then there may have some _1s‘ in the background which is called as background noise, Also, there is a possibility that system generated an error in recognizing gesture this may be termed as gesture noise. If we want flawless contour detection of a gesture, then abovementioned errors should be nullified. A morphological separating (filtering) approach is employed utilizing grouping of dilation (enlargement) and erosion (disintegration) to accomplish a smooth, shut, and finish the contours of a hand motion.

6.1.4 Extraction of Feature

Pre-prepared or preprocessed picture is accessible to be utilized and different highlights of the resultant picture are removed. Following are the features that can be extracted:

i. Finding Contours
ii. Finding and correcting convex hull
iii. Mathematical Operations

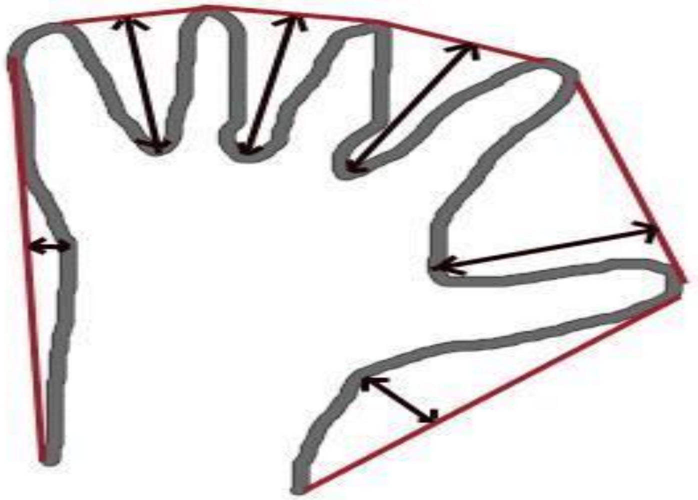


Figure 6.4: Feature Extraction

1. Contours: It implies the direction of hand i.e. regardless of whether the hand is on a horizontal plane or vertically set. Initially, we try to find the orientation by length to width ratio with a presumption that if the hand is vertical ten length of the box bounding

them will be more than the width of the same box and, if hand is horizontally placed then width of bounding box is larger than width of the box binding the hand will be more than that of the length of the box.

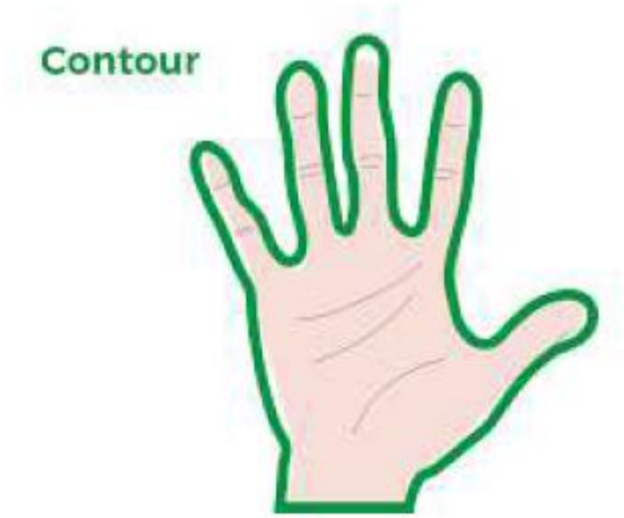


Figure 6.5: Detected Contour of the image

2. Finding and correcting convex hulls: A hand posture is recognized by its own orientation and by the fact, that how many fingers are shown. For getting the aggregate of how many fingers are shown in hand motions that we have to process just area of the finger of the hand that we have in past advance by figuring out and analyzing the centroid.

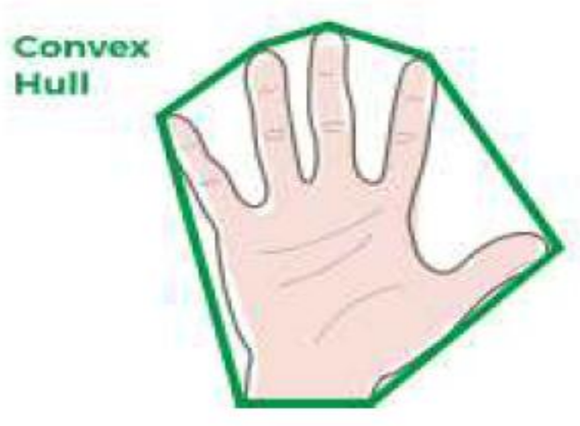


Figure 6.6: Detected Convex Hull

3. Math Operations: This can be calculated by - angle = math.acos ((b**2 + c**2 - a**2)/(2*b*c)) * 57 this formula determines the angles between the two fingers, for distinguish between the different fingers and to identify them all. We can also determine the length of each raised or collapsed finger coordinate points taking centroid as a source of perspective point, this can be done keeping in mind the end goal to extricate the correct number of finger brought up in the picture.

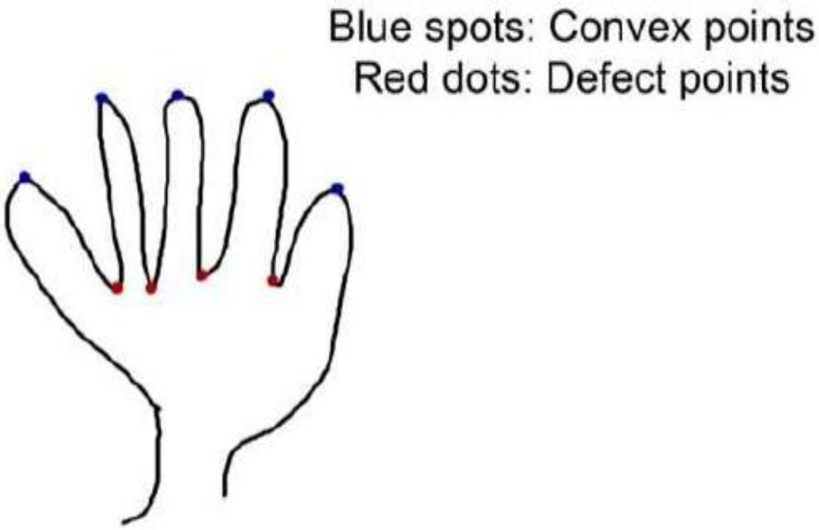


Figure 6.7: Detected convex and defect points in the image.

6.2 Implementation Code

```
# integrated code for the application with required comments for explanation

import numpy as np

import cv2

import math

import pyautogui

# Open Camera

capture = cv2.VideoCapture(0)
```

```
while capture.isOpened():
 # Capture frames from the camera
 ret, frame = capture.read()
 # Get hand data from the rectangle sub window
  cv2.rectangle(frame, (100, 100), (300, 300), (0, 255, 0), 0)
  crop_image = frame[100:300, 100:300]
  # Apply Gaussian blur
   blur = cv2.GaussianBlur(crop_image, (3, 3), 0)
   # Change color-space from BGR -> HSV
    hsv = cv2.cvtColor(blur, cv2.COLOR_BGR2HSV)
    # Create a binary image with where white will be skin colors and rest is black
    mask2 = cv2.inRange(hsv, np.array([2, 0, 0]), np.array([20, 255, 255]))
    # Kernel for morphological transformation
    kernel = np.ones((5, 5))
    # Apply morphological transformations to filter out the background noise
    dilation = cv2.dilate(mask2, kernel, iterations=1)
    erosion = cv2.erode(dilation, kernel, iterations=1)
     # Apply Gaussian Blur and Threshold
    filtered = cv2.GaussianBlur(erosion, (3, 3), 0)
    ret, thresh = cv2.threshold(filtered, 127, 255, 0)
    # Show threshold image
    cv2.imshow("Thresholded", thresh)
```

```
# Find contours
 contours, hierarchy = cv2.findContours(thresh.copy(), cv2.RETR_TREE,
 cv2.CHAIN_APPROX_SIMPLE)
  try:
  # Find contour with maximum area
  contour = max(contours, key=lambda x: cv2.contourArea(x))
  # Create bounding rectangle around the contour
   x, y, w, h = cv2.boundingRect(contour)
   cv2.rectangle(crop_image, (x, y), (x + w, y + h), (0, 0, 255), 0)
   # Find convex hull
    hull = cv2.convexHull(contour)
   # Draw contour
    drawing = np.zeros(crop_image.shape, np.uint8)
    cv2.drawContours(drawing, [contour], -1, (0, 255, 0), 0)
    cv2.drawContours(drawing, [hull], -1, (0, 0, 255), 0)
    # Find convexity defects
    hull = cv2.convexHull(contour, returnPoints=False)
    defects = cv2.convexityDefects(contour, hull)
    # Use cosine rule to find angle of the far point from the start and end point i.e. the
    # Convex  points (the finger tips) for all defects
      count_defects = 0
    for i in range(defects.shape[0]):
        s, e, f, d = defects[i, 0]
        start = tuple(contour[s][0])

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

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

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

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

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

        angle = (math.acos((b ** 2 + c ** 2 - a ** 2) / (2 * b * c)) * 180) / 3.14

        # if angle > 90 draw a circle at the far point

        if angle <= 90:

            count_defects += 1

            cv2.circle(crop_image, far, 1, [0, 0, 255], -1)

            cv2.line(crop_image, start, end, [0, 255, 0], 2)

# Print number of fingers

 if count_defects == 0:

        cv2.putText(frame, "PLAY", (50, 50), cv2.FONT_HERSHEY_SIMPLEX,

         2,(0,0,255),2)

        pyautogui.click((pyautogui.locateCenterOnScreen('play1.png')))

  elif count_defects == 1:

        cv2.putText(frame, "PAUSE", (50, 50), cv2.FONT_HERSHEY_SIMPLEX,

         2,(0,0,255), 2)

        pyautogui.click((pyautogui.locateCenterOnScreen('pause.png')))

    elif count_defects == 2:

        cv2.putText(frame, "FORWARD", (5, 50), cv2.FONT_HERSHEY_SIMPLEX,
        2,(0,0,255), 2)
         pyautogui.click((pyautogui.locateCenterOnScreen('forward.png')))
    elif count_defects == 3:
        cv2.putText(frame, "BACKWARD", (50, 50), cv2.FONT_HERSHEY_SIMPLEX,
        2,(0,0,255), 2)
        pyautogui.click((pyautogui.locateCenterOnScreen('backward.png')))
    elif count_defects == 4:
        cv2.putText(frame, "STOP", (50, 50), cv2.FONT_HERSHEY_SIMPLEX, 2,(0,0,255),
         2)
        pyautogui.click((pyautogui.locateCenterOnScreen('stop.png')))
    else:
        pass
# Show required images
cv2.imshow("Gesture", frame)
all_image = np.hstack((drawing, crop_image))
cv2.imshow('Contours', all_image)
# Close the camera if 'q' is pressed
if cv2.waitKey(1) == ord('q'):  break
capture.release()
cv2.destroyAllWindows()
```

CHAPTER 7

RESULTS AND DISCUSSION

Here, what we do is, we just open our script file it will automatically launch a video player. Here we have chosen VLC Media Player. Then script stops execution for pre-defined time to load the media player. After video file is being played then system invokes the tools that we required to run it for instance- OpenCV, Camera, pyautogui. Now, we are ready to do just sit back and control without using any conventinal method. By, pointing out finger in plain background, we can get following output.

NO. OF FINGERS	*ACTION - PERFOMED*
0	**Play**
1	**Pause**
2	**Seek Backward**
3	**Seek Forward**
4	**Stop**

In real time by using web camera the input video is taken and converted into frames than some of the steps are carried out as shown in the figures to count the number of fingers. The experimental results are shown below:

1. The Procured image is RGB and must be processed before i.e. pre-processed before the components are separated and acknowledgement is made and is shown in figure 7.1.

2. We in this project have used Otsu’s thresholding technique. Otsu‘s thresholding is used to automatically perform cluster-based thresholding . Thresholding techniques are employed in partitioning the image pixel histogram by using a single threshold technique.

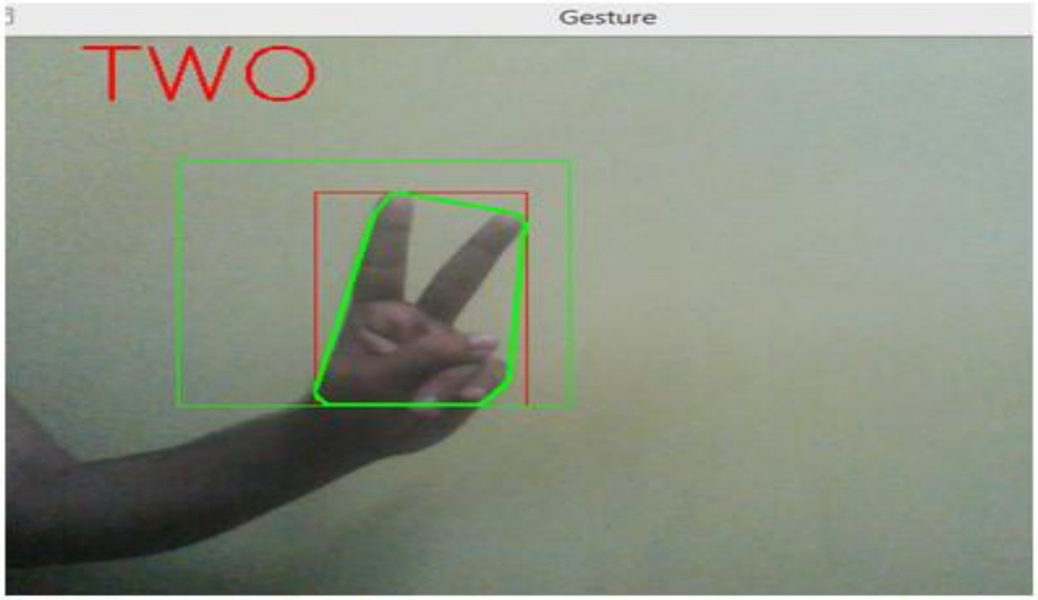


Figure 7.1: Hand Gesture(RGB Image)

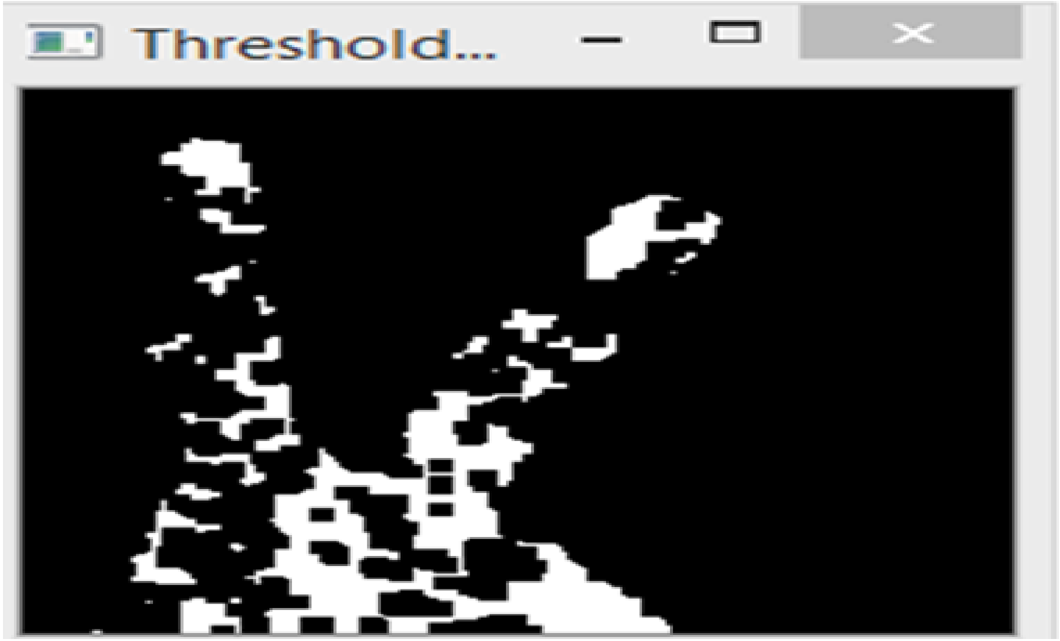


Figure 7.2: Threshold Image(Grey Scale Image)

3. Contours square measure the curves change of integrity all the continual points on the boundary, having same color or intensity. The contours square measure a great tool for form analysis and object detection and recognition. The contour is drawn on the boundary of the hand image that is found once thresholding.

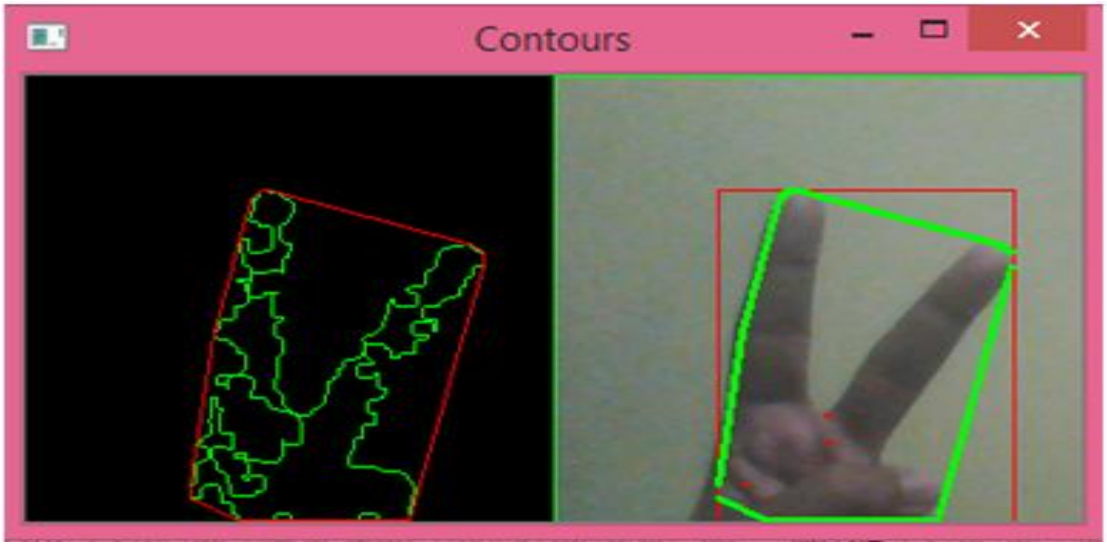


Figure 7.3: Contour Detection

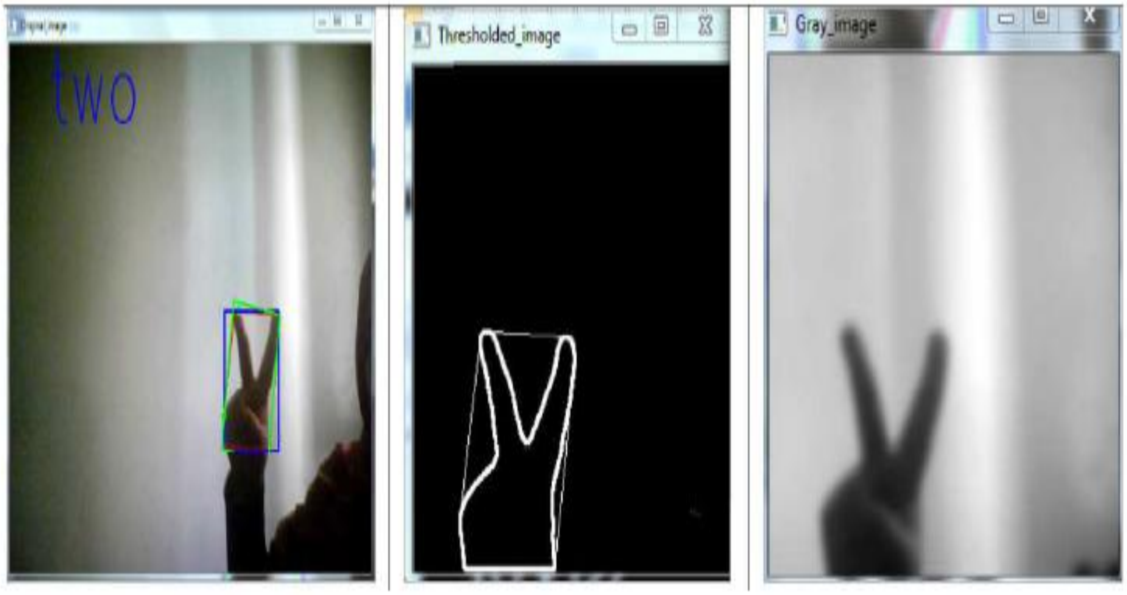


Figure 7.4: Show input frame count two and threshold image and gray image

4. The proposed system uses hand gesture, mostly no of fingers raised within the region of Interest to perform various operations. A Hand Gesture Recognition System recognizes the Shapes and or orientation depending on implementation to task the system into performing some job as shown in figure 7.5.

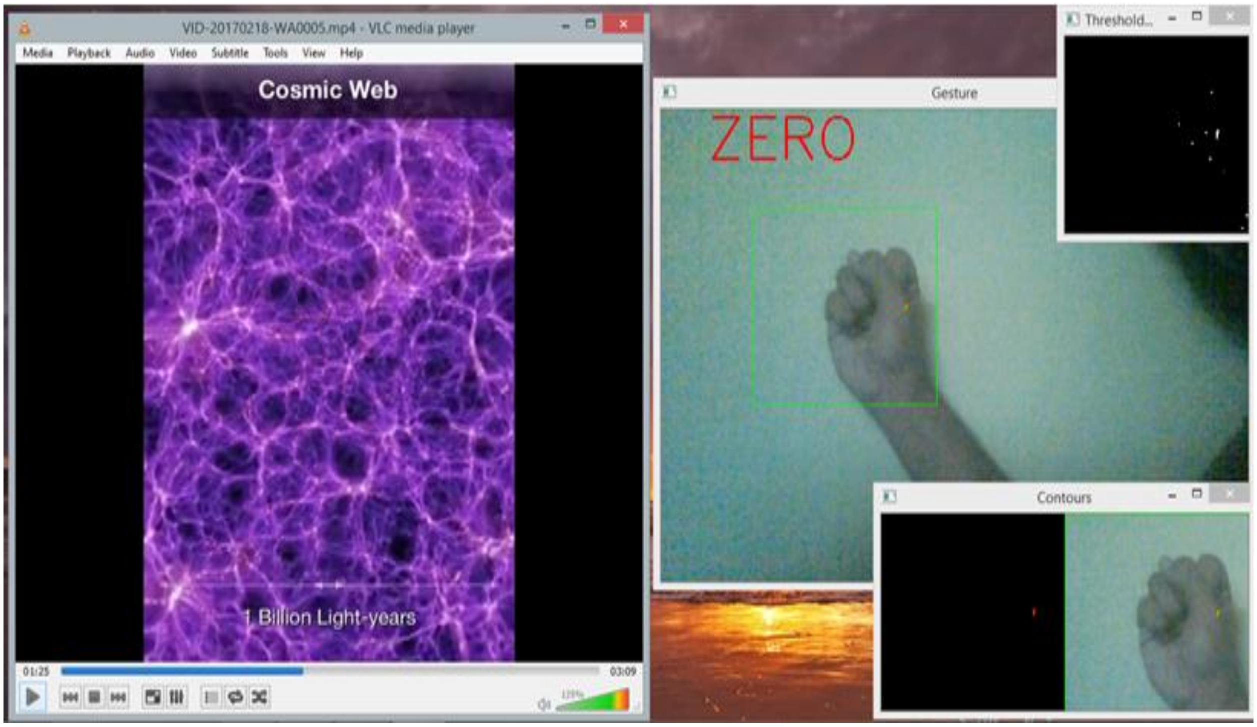


Figure 7.5: Proposed System Application(VLC media Player)

CHAPTER 8

TESTING

Hand Gestures Recognition System testing is actually a series of different tests whose primary purpose is to fully exercise the computer-based system. Although each test has a different purpose, all work to verify that all the system elements have been properly integrated and perform allocated functions. The testing process is actually carried out to make sure that the product exactly does the same thing what is supposed to do. In the testing stage following goals are tried to achieve:-

□ To affirm the quality of the project.

□ To find and eliminate any residual errors from previous stages.

□ To validate the software as a solution to the original problem.

□ To provide operational reliability of the system.

8.1 Quality Assurance

Quality assurance consists of the auditing and reporting functions of management. The goal of quality assurance in hand gesture recognition system is to provide management with the data necessary to be informed about product quality, thereby gaining insight and confident that the product quality is meeting its goals. This is an “umbrella activity” that is applied throughout the engineering process**.** Software quality assurance encompasses:-

□ Analysis the number of finger raised

□ Formal technical reviews that are applied during each software engineering

□ Feature such as contour and convex hull to be extracted without any distortion

□ Control of software documentation and the change made to it.

□ A procedure to ensure compliance with software development standards.

□ Measurement and reporting mechanisms of different feature pre-processed.

8.1.1 Quality Factors

An important objective of quality assurance is to track the software quality and assess the impact of methodological and procedural changes on improved software quality. The factors that affect the quality can be categorized into two broad groups:

□ Factors that can be directly measured i.e. the thresholding value for the binary image.

□ Factors that can be indirectly measured i.e. contours and convexity defects points etc

8.2 Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centred on the following items:

Valid Input : Number of fingers raised.

Invalid Input : Number of finger raised with blurred pixel or wrong number of finger raised.

Functions : identified functions for pre-processing and feature extraction

Output : Automated controlling of Media player .

Systems/Procedures: Interfacing systems or procedures must be invoked for pre-processing.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing.

CHAPTER 9

CONCLUSION & FUTURE SCOPE

9.1 CONCLUSION

The proposed system is a real time video processing that is based on a real time application system. This can replace one of the traditionally used input device i.e. mouse so that simply by using the hand gestures the user will be able to interact naturally with their computer.

In this project we have planned, designed and implemented the system for Hand gesture recognition system for controlling UI which is a standalone application for controlling the various user interface controls and/or programs like VLC Media Player. In the analysis phase we gathered information regarding various gesture recognition systems existent today and the techniques and algorithms they employ and the success/failure rate of these systems. Accordingly, we made a detailed comparison of these systems and analyzed their efficiency. In the design phase we designed the system architecture diagrams and also the data flow diagram of the system. We studied and analyzed the different phases involved and accordingly designed and studied the algorithms to be used for the same.

With the help of observations that we have, we can conclude that the results should be depend upon:

- The is the concept of finding contours while converting the gray image to the binary image as the concept of thresholding states. For instance, the lighting over the photo of the hand possibly uneven which is caused by drawing shapes of the contours which is around the dim regions notwithstanding the form around the hand. Changing the limit should be kept that from happening.
- Background of the pictures should be plain to get accurate analysis of recognition of gestures.

- Extra occasional keep an eye on minutes is helpful for checking whether the forms of the layout picture and the picture of individual have same shape.

9.2 FUTURE SCOPE

- This project can be more interactive with the help of tracking real time hand movements and controlling mouse pointer on screen. The shortcoming of requiring a plain background can be overcome with the help of Background Image substraction or Machine Learning Techniques.
- To create a website which operates using hand gestures. JavaScript can be dynamically combined with the gesture recognition logic for the same.
- To use the gesture recognition logic in sensitive areas of work like hospitals and nuclear power plants where sterility between machines and human is vital.
- To create a battery free technology that enables the operation of mobile devices with hand gestures.

REFERENCES

[1]: S. Padmappriya & K. Sumalatha – Digital Image Processing Real Time Application;International Journal Of Engineering Science Invention(IJESI),2018

[2]: Henrik Birk and Thomas Baltzer Moeslund, ―Recognizing Gestures from the Hand Alphabet Using Principal Component Analysis‖, Master‗s Thesis, Laboratory of Image Analysis, Aalborg University, Denmark, 1996.

[3]: Mr. Deepak K. Ray, Mayank Soni –Hand Gesture Recognition using Python / International Journal on Computer Science and Engineering (IJCSE) . Assistant Professor: MCA Dept , ICEM, Parandwadi Pune, India

[4]: Prof. Praveen D. Hasalkar1, Rohit S. Chougule2, Vrushabh B. Madake3, Vishal S. Magdum-Hand Gesture Recognition System, / *International Journal of Advanced Research in Computer and Communication Engineering*, Department of Computer Science and Engineering, W.I.T, Solapur, Maharashtra, India.

[5]: Human Hand Gesture Recognition - *Usama Sayed* 1, *Mahmoud A. Mofaddel* 2, *Samy Bakhee t* 2 *and Zenab El-Zohry* 2,_1 Department of Electrical Engineering, Faculty of Engineering, Assiut University, P.O. Box 71516 Assiut, Egypt; 2 Department of Math and Computer Science, Faculty of Science, Sohag University, P.O. Box 82524 Sohag, Egypt.