

CHAPTER-01

SOFTWARE

REQUIREMENT

SPECIFICATION

1.1 INTRODUCTION

The essential aim of building hand gesture recognition system is to create a natural interaction between human and computer where the recognized gestures can be used for controlling a robot or conveying meaningful information. Gestures can originate from any bodily motion or state but commonly originate from the face or hand. Many approaches have been made using cameras and computer vision algorithms to interpret sign language.[1]

A gesture is a spatiotemporal pattern which may be static , dynamic or both, and is a form of non-verbal communication in which bodily motions convey information. Gestures include motion of head, hands, fingers or other body parts. Gesture Recognition collectively refers to the whole process of tracking human gestures , to their representation and conversion to semantically meaningful commands.[2] Gesture Recognition and more specifically hand gesture recognition can be used to enhance Human Computer Interaction (HCI) and improve the effective utilisation of the available information flow.

Digital Cameras are now integrated into personal computers, mobile cellular devices and handheld systems. These devices usually include a powerful microprocessor, capable of performing millions of computations per second.[3] The technology on digital cameras and microprocessors are advancing rapidly that it is possible to create a human computer interfaces using these resources for recognition of user gestures. The Gesture recognition interface acts as a communication channel between humans and machines. The human-machine interaction is similar to human-human interaction, in which, the valuable information are communicated using the human organs like hand gesture, head movement, face expression, voice communication and overall body posture.[4]

The design of a gesture recognition system should be based on common hardware support such as web-cams or mobile-integrated cameras, to be applicable to current PCs, mobile devices, Digital Cameras, etc. While designing such systems, certain parameters have to be included, so that the system will be able to operate under complex or non-uniform background, i.e., different light intensity and noisy environment, etc.[2]

1.1.1 Purpose

The aim of the project is to create a software that recognises pre defined hand gestures using various computer vision and machine learning algorithms. As mentioned in figure 1.2 we can divide the project into three major steps which represent the major objectives in the project.[5]

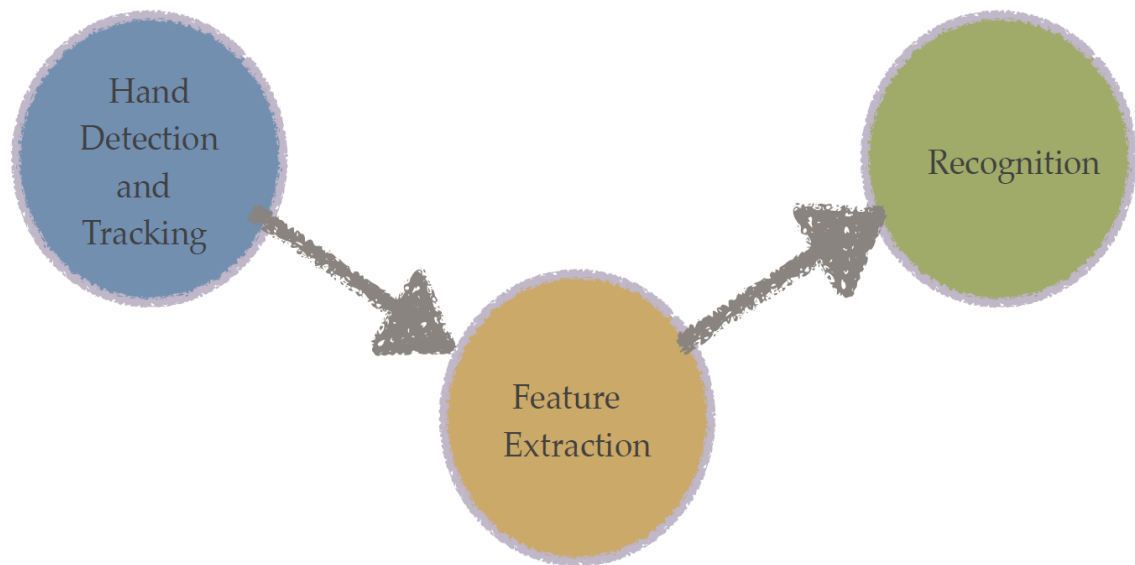


Figure 1: Steps For Hand Gesture Recognition

Hand Detection and Tracking

This step deals with detection of hand in the frame and tracking it through the video, our objective in this step is to create a robust system that can detect and track hands of different skin colours in varying light conditions with different but simple background.[6]

Feature Extraction

This step deals with extracting important features that represent important characteristics of the gesture throughout the video and then storing these features. Our objective in this step is to find features that represent shape, motion, size, reflectivity and other important properties.[7] We want features that are generative and not discriminative as this will allow multiple gestures to be recognised with limited features.

Recognition

This step deals with recognising and classifying the performed gesture. It has two phases, the training phase which involves training the system on datasets and the classification phase which involves classifying the performed gestures, our objective in this step is to obtain classification with high accuracy within minimum time.[3]

1.1.2 Project Scope

The scope of this project is to build a real time gesture classification system that can automatically detect gestures in natural lighting condition. In order to accomplish this objective, a real time gesture based system is developed to identify gestures.

This system will work as one of futuristic of Artificial Intelligence and computer vision with user interface. Its create method to recognize hand gesture based on different parameters. [6] The main priority of this system is to simple, easy and user friendly without making any special hardware. All computation will occur on single PC or workstation. Only special hardware will use to digitize the image (Digital Camera).

Human Computer Interaction is still in its infancy. Visual interpretation of hand gestures today allows the development of potentially natural interfaces to computer-controlled environments. Though most current systems employ hand gestures for manipulation of objects the complexity of the interpretation of gestures dictates the achievable solution. Hand gestures for HCI are mostly restricted to single handed and produced by single user in the system. This consequently downgrades the effectiveness of the interaction. Computer Vision methods for hand gesture interfaces must surpass current performance in terms of robustness and speed to achieve interactivity and usability. Considering the relative infancy of research related to vision based gesture recognition remarkable progress has been made. [8]

1.1.3 Operating Environment

Open CV

OpenCv 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.

OpenCv was started 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.[9]

The basic tutorial documentation is provided by Bradsky and Kaehler. 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). [5]

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 colors per pixel
- depth – an integer showing the number of bits per pixel
- widthStep – an integer showing the number of bytes per image row
- imageSize – 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,
- floodfill 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 subregions (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/ naive Bayes classifier,
- Decision trees classifier,
- Boosting group of classifiers,
- Neural networks algorithm, and
- Support vector machine classifier.

Visual Studio 2017

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs for Microsoft Windows, as well as web sites, web apps, web services and mobile apps. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. It can produce both native code and managed code.

Visual Studio includes a code editor supporting IntelliSense (the code completion component) as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger. Other built-in tools include a code profiler, forms designer for building GUI applications, web designer, class designer, and database schema designer. It accepts plug-ins that enhance the functionality at almost every level—including adding support for source control systems (like Subversion) and adding new toolsets like editors and visual designers for domain-specific languages or toolsets for other aspects of the software development lifecycle (like the Team Foundation Server client: Team Explorer).

Visual Studio supports 36 different programming languages and allows the code editor and debugger to support (to varying degrees) nearly any programming language, provided a language-specific service exists. Built-in languages include C,[6] C++ and C++/CLI (via Visual C++), VB.NET (via Visual Basic .NET), C# (via Visual C#), F# (as of Visual Studio 2010[7]) and TypeScript (as of Visual Studio 2013 Update 2). Support for other languages such as Python,[8] Ruby, Node.js, and M among others is available via language services installed separately. It also supports XML/XSLT, HTML/XHTML, JavaScript and CSS. Java (and J#) were supported in the past.

Microsoft provides a free version of Visual Studio called the Community edition that supports plugins and is available at no cost.

Setting up the environment:

1. Install and configure OpenCV: I used the first method and installed it using the pre-built libraries in C:\
2. Install Visual Studio 2017.
3. Open Visual Studio and start a new project. Choose Win32 console application and name it appropriately.(Yes, even for 64 bit applications. Apparently 32 in Win32 does not refer to it's bit-ness.for more details refer this.)
4. In the create project wizard , select empty project in the second window of the wizard and keep everything else as it is.
5. Right click on project name and open properties->configuration->manager->platform->new->x64->ok->close(this will make the application run in 64 bit systems)
6. Including C/C++ opencv libraries:
C/C++ -> General->Additional Include Directories ->edit -> double click empty area -> click .. -> browse to C:\opencv\build\include
similarly include:
C:\opencv\build\include\opencv
and
C:\opencv\build\include\opencv2
Click OK
7. Including Linker Libraries:
Linker->General->Additional Library Directories->edit->browse and select
C:\opencv\build\x64\lib
Linker->Input->Additional Dependencies->edit->copy the names of files with d in the end. in m case it was:
opencv_ts300d.lib
opencv_world300d.lib
Click OK
8. Add OpenCV path to system environment: C:\opencv\build\x64\vc12\bin
9. Right Click Source files under the project->name->add->new item->C++ File

Software Requirements: A set of instructions or program required to make hardware platform suitable for desired task is known as software. Software can also be defined as the utility programs that are required to drive hardware of computer.

- Operating system- Microsoft Windows 7 SP 1 or above
- Microsoft Visual Studio 2017
- Visual C++ compilers (for Windows)
- Supporting Webcam Drivers 4.2

Hardware Requirements: All the physical equipment's i.e. input devices, processor, and output device & inter connecting processor of the computer s called as hardware.

- Hard Disk minimum of 40 GB.
- RAM minimum of 2 GB.
- Dual Core and up
- 15" Monitor
- Integrated webcam or external webcam (15 -20 fps).

1.1.4 Methodology

PROPOSED METHODOLOGY

This project basically deals with the design of a system that acquires a user's hand gesture and classifies it based on web cam in a run-time environment through openCV library function and different optimize algorithms.

The system will include low-resolution web cam for capturing the hand gestures and an algorithm that processes the acquired images and then classifies the hand gesture correctly. The work mainly emphasizes on the feature extraction from the hand gestures and use that features in the recognition algorithms.[9] Initially, the system will contain a setup procedure, in which, the algorithm is trained on given training set, based on significant feature extracted for different hand gestures.

Once the setup is completed successfully, the system will be able to classify the given hand gesture based on the knowledge acquired during the training phase.

METHOD TO DESIGN

The design of hand gesture recognition system is broadly divided into two phase.

- The first phase is the preprocessing phase.
- The second phase is the classification phase.

PREPROCESSING PHASE

The efficiency of the Classification phase entirely depends on the preprocessing phase i.e., better the task performed in pre-processing phase, better will be the performance of classification phase. So, all the tasks in pre-processing phase are to be carried out properly.[6]

The main purpose of the pre-processing stage is to:

- Extract the only hand gesture from an image.
- Remove the noises (if present) and unwanted region.
- Process the extracted image to form a binary image and
- Extract the distinguishable significant features from the processed image, to form a feature set for classification.

CLASSIFICATION PHASE

Based on the Research paper, for classification, there are lots of efficient algorithms that are already used. They are Gradient, PCA (Principal Component Analysis) and SVM (Support Vector Machine). But, the paper mainly focuses on SVM, which is a machine learning algorithm. Initially, the traditional methods will be followed in preprocessing step, for preparing an image for feature extraction. Once a prepared image (i.e. noise free image), is successfully achieved, the significant features representing a different hand gestures are extracted and represented to include in the classification algorithm. [3]. This is because the more features we include in the algorithm the more will be the accuracy of classification.



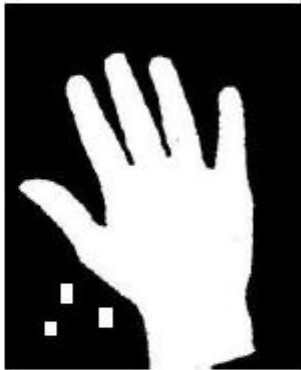



Process	Input	Output
RGB to Binary		
Noise Removal		
Skeletonization		

TABLE 1: Contour Extraction Process

EXPERIMENTAL SETUP FOR IMPLEMENTATION

As stated earlier, the set up procedure for a design of a system was basically divided into two parts, preprocessing and classification part. However, the preprocessing part plays an important role for classification. The first step in pre-processing part is the image acquisition step, in which, the scene containing hand portion as a major object was stored. During image acquisition, a particular constrain is included in a system, i.e., the images for training as well as testing were taken at equidistant level. Once an image containing hand portion only is stored, the second step in preprocessing was to extract the hand portion only from uniform or non-uniform background.[8]

WEB CAMERA

The purpose of Web camera is to capture the human generated hand gesture and store its image in memory. The package called Java Media Framework is used for storing image in memory and again calling the same program after particular interval.

FEATURE EXTRACTION

This task for the extracting the hand portion was accomplished by considering the certain range of pixel values that represents skin color of hand. In this process an assumption is made that is, the color of background or the color of objects in background should not be a similar to that skin color.[1]

To reduce the complexity of feature extraction for hand gestures, the output image of hand portion extraction process was converted into binary image using a thresholding technique.

$$\bar{X} = \frac{\sum_{i=0}^k x_i}{k}, \quad \bar{Y} = \frac{\sum_{i=0}^k y_i}{k}$$

OPTIMIZE IMAGE

The image so formed may contain some noises. The appearance of such noises may be due to the atmospheric condition at which the images were taken and also the type of source that is used for capturing an image. Hence to remove the noise, the median filter was generally used. Also, depending upon the percentage of noise present in the binary image, it was found that the morphological operator namely image erosion can also be

used for remove small sharp unwanted details (i.e., noise) from an image. The extent of noise removal is directly proportional to the extent to which a system can be trained correctly and hence classifies the input hand gestures correctly.[10]

The next immediate process after the removal of noise is the feature extraction process, in which, the different techniques are applied based on the type of feature to be extracted. There are various different kinds of distinguished features that can be extracted from the filtered image, but the paper focuses only on the active and in-active finger which is represented by 1 and 0 respectively.

$$\text{Distance} = \sqrt{(x_2 - x_1)^2 - (y_2 - y_1)^2}$$

$$M1 = (\eta_{20} + \eta_{02}),$$

$$M2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2,$$

$$M3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2,$$

$$M4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2,$$

$$M5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03}) \\ [3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2],$$

$$M6 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}),$$

$$M7 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] - (\eta_{30} + 3\eta_{12})(\eta_{21} + \eta_{03}) \\ [3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2].$$

DETECTING PALM

So to identify the active and in-active finger, the first task was to identify the finger tip. To identify the finger tip, the skeletonization technique was used to thin the finger portion of hand. In this technique, the thinned image was obtained that represents the finger portion only by singly connected pixels, the co-ordinate values for the finger tip was stored. The pixel is considered as a finger tip that has only one neighbor in 3x3 window mask. For such pixel, the co-ordinate values are stored for different active finger. After storing the co-ordinate values of different active finger tip.[11]

The next step is to find the centroid of a hand. In the next step, the distance between the centroid and the finger tip was calculated. To increase the efficiency of the system, a certain degree of deviation was included in the distance parameter, so that a system can

recognize the hand gesture correctly even there is a small bents of finger in the gesture to be recognized.[7]

IMAGE PROCESSING ALGORITHM

This carries the major portion of implementation. First the captured image is preprocessed by techniques like real-time hand tracking and extraction algorithm, feature extraction, hidden markov model(hmm) training and gesture recognition.[4]

EVENT HANDLING

Once the gesture is identified the appropriate command for it will be executed. This includes controlling mouse, performing its various applications like selecting, dragging and pasting any folder from one place to another, both left and right clicks and scrolling. [1]. This also includes controlling all function such as traffic signals through different gestures for each signal.

AMERICAN SIGN LANGUAGE (ASL)

American Sign Language (ASL, also Ameslan) is the dominant sign language of the Deaf community in the United States, in the English-speaking parts of Canada, and in parts of Mexico. “It is a manual language or visual language, meaning that the information is expressed not with combinations of sounds but with combinations of handshakes, palm orientations, movements of the hands, arms and body, and facial expressions.”[12]

The main project task is to produce a gesture recognition system, the system will be able to recognize the ASL gesture using web cameras. In the real world, it has practical usage, if the mute demonstrates the hand posture in front of the web-camera, the system will detect which letter it is in the ASL; If we know which letter it is, we will understand which word several gesture represents, it will help mute people to communicate with normal people more easily.

1.2 FLOWCHARTS

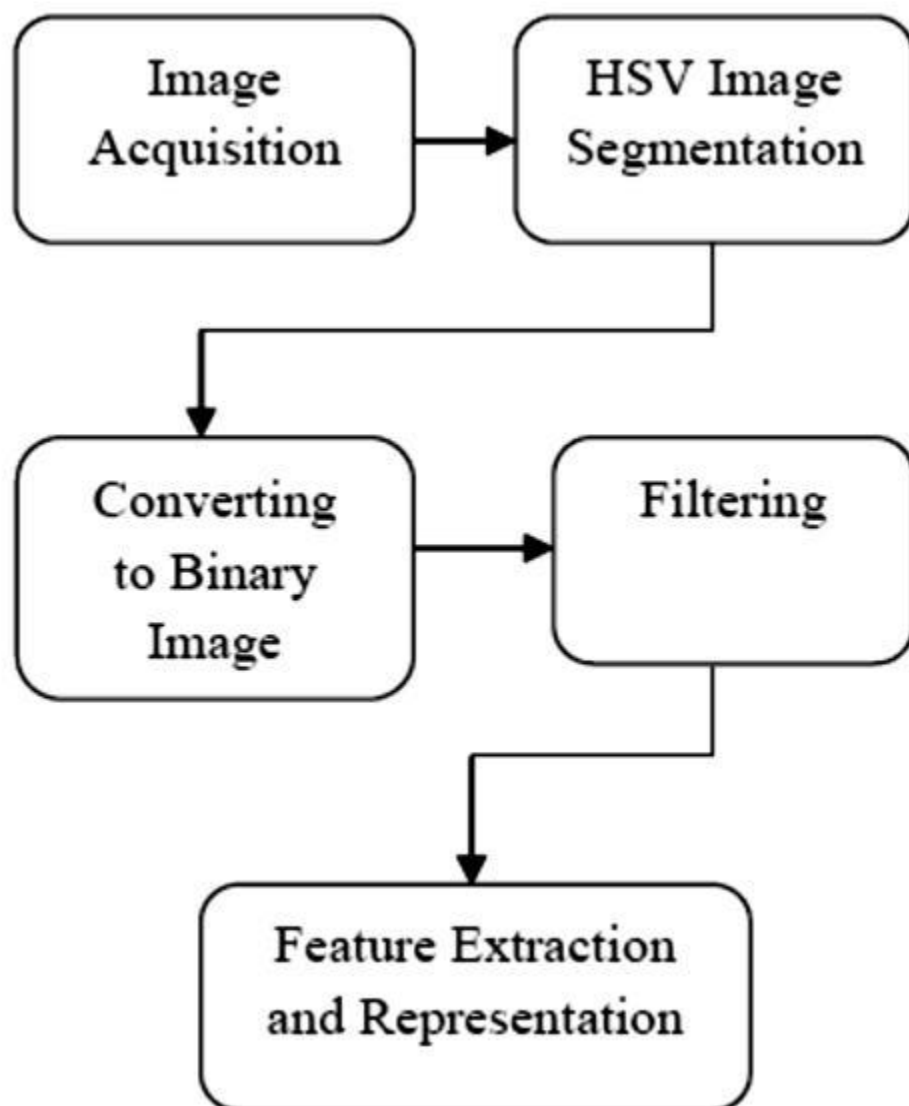


Figure 2: Pre-processing Phase

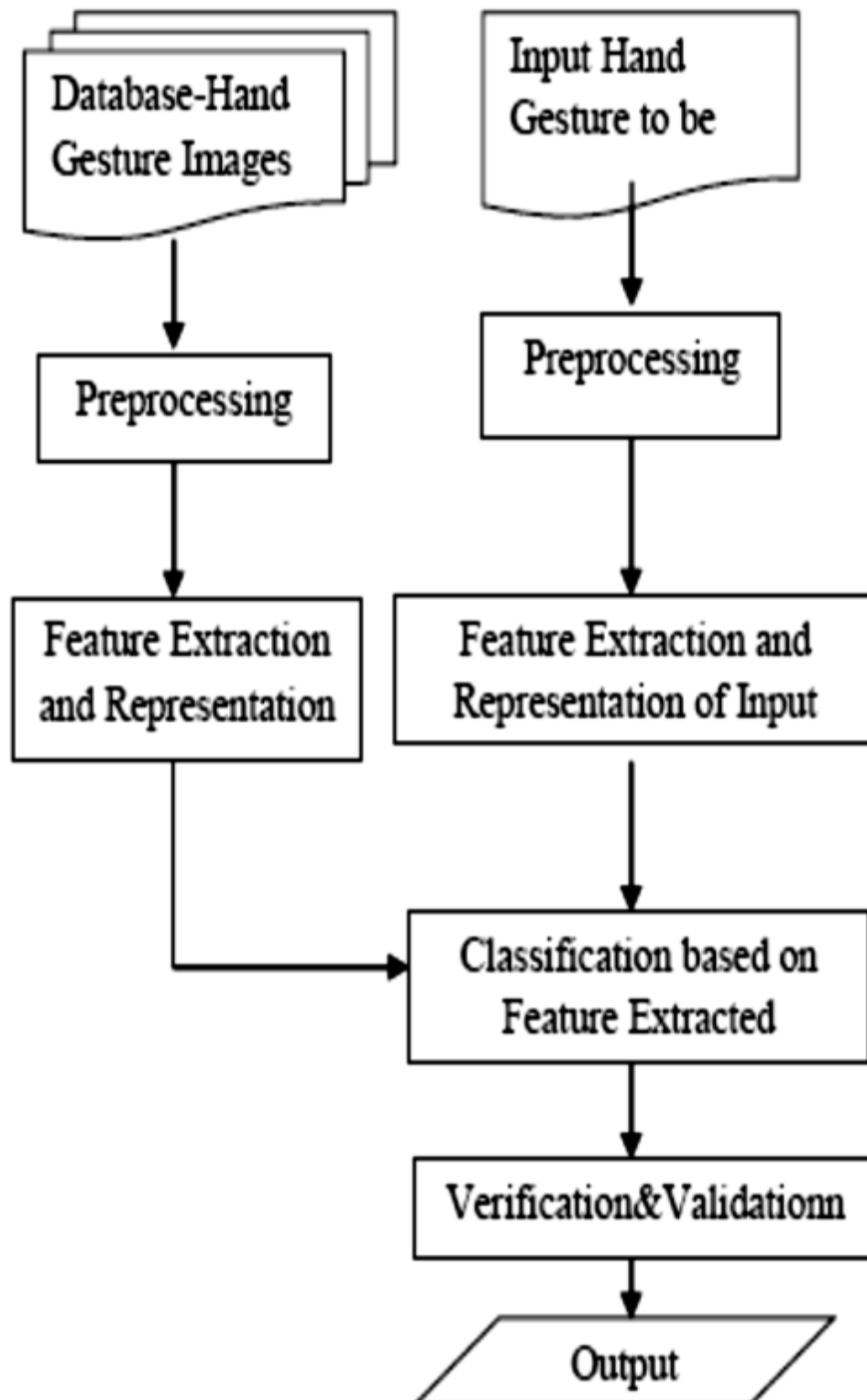


Figure 3: General flow diagram of a System

1.3 PROBLEM STATEMENT

Nowadays, gestures still are naturally used by many people and especially are the most major and nature interaction way for deaf people . In recent years, the gesture control technique has become a new developmental trend for many human-based electronics products, such as computers, televisions, and games.

“Hand Tracking And Gesture Recognition ” is based on concept of Image processing. In recent year there is lot of research on gesture recognition using kinect sensor on using HD camera but camera and kinect sensors are more costly. This paper is focus on reduce cost and improve robustness of the proposed system using simple web camera

1.4 REASON TO CHOOSE THE PROBLEM

Gestures are a major form of human communication. Hence gestures are found to be an appealing way to interact with computers, as they are already a natural part of how we communicate. A primary goal of gesture recognition is to create a system which can identify specific human gestures and use them to convey information for device control and by implementing real time gesture recognition a user can control a computer by doing a specific gesture in front of a video camera linked to the computer.[6] A primary goal of gesture recognition research is to create a system which can identify specific human gestures and use them to convey information or for device control. This project covers various issues like what are gesture, their classification, their role in implementing a gesture recognition system, system architecture concepts for implementing a gesture recognition system, major issues involved in implementing a simplified gesture recognition system, exploitation of gestures in experimental systems, importance of gesture recognition system, real time applications and future scope of gesture recognition system

CHAPTER-02

LITERATURE

REVIEW

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 and computers, man has transcending many visual limitations. He can see into far galaxies, the microscopic world, the sub-atomic world, and even “observe” infra-red, x-ray, ultraviolet and other spectra for medical diagnosis, meteorology, surveillance, and military uses, all with great success.[2]

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 designed to analyze and acquire information from images autonomously in the same natural way humans can. According to Gonzales and Woods [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 [3]. 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 decades-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 both draw the proverbial water Real-Time Hand Gesture Detection and Recognition Using Simple Heuristic Rules 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 the purpose of producing another desired image as the end result. For example, the blurred image of a car registration plate might be enhanced by imaging techniques to produce a clear photo of the same so the police might identify the owner of the car.[10] On the other hand, computer vision takes a digital image and subjects it to the same digital imaging techniques but for the purpose of analyzing 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 distill 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 considered 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. Here computer vision uses the techniques of artificial intelligence in knowledge representation to enable a “classifier” to match classes to objects based on the pattern of their features or structural descriptions. A classifier has to “learn” the patterns by being fed a training set of objects and their classes and achieving the goal of minimizing mistakes and maximizing successes through a step-by-step process of improvement.[7]

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 at the moment relies heavily upon the science and technology of artificial intelligence.[9]

2.2 OpenCV

OpenCV (Open Source Computer Vision) is a library of programming functions mainly aimed at real-time computer vision.[2] Originally developed by Intel, it was later supported by Willow Garage and is now maintained by Itseez.[1] The library is cross-platform and free for use under the open-source BSD license.

OpenCV supports the Deep Learning frameworks TensorFlow, Torch/PyTorch and Caffe. OpenCv 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.

OpenCv was started 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.

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 colors per pixel
- depth – an integer showing the number of bits per pixel
- widthStep – an integer showing the number of bytes per image row
- imageSize – an integer showing the size of in bytes

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

- Normal/ naive 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 derived. A *feature* is any characteristic of an image, or any region within it, that can be measured. Objects with common features may be grouped into classes, where the combination of features may be considered a *pattern*. Object recognition 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*.^[9]

The general steps in pattern recognition may be summarized in Figure 1 below:

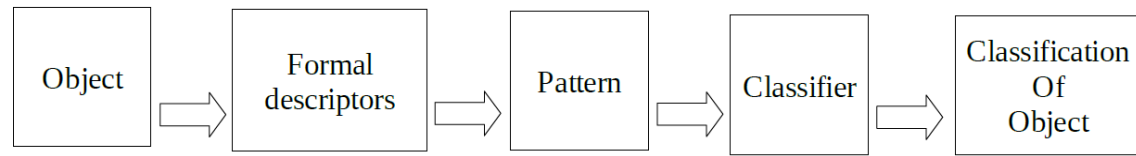


Figure 4: General pattern recognition steps

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 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.[15]

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, and the set of the feature vectors (objects) inputs and corresponding outputs of classifications (both positive and negative results) is called the *training set*

Sno.	Binary Code	Symbol
1	00001	A
2	00100	B
3	00110	C
4	10010	D
5	10011	E
6	01100	F
7	01010	G
8	10100	H
9	11000	I
10	01110	J
11	11100	K
12	11010	L
13	10110	M
14	11110	N
15	00011	O
16	00111	P
17	01111	Q
18	11111	R
19	11011	S
20	10111	T
21	11101	U
22	11001	V
23	10101	W
24	00101	X
25	00010	Y
26	00000	Z

TABLE 2: Lookup Table For Characters

2.4 Moment Invariants

As mentioned previously, 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.[3]

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. [11]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. The concept of statistical characteristics or moments that would be indifferent to geometric transformations was first formulated by Hu in 1962. Moments are polynomials of increasing order that describe the shape of a statistical distribution [10]. The order of a moment is indicated by its exponent. The geometric moments of different orders represent different spatial characteristics of the image intensity distribution.

CHAPTER-03

PLANNING

AND

SPECIFICATION

There are several approaches that have been used to design a hand gesture recognition system. In all the approaches, the primary focus was given to feature extraction of the hand gesture and it was found that the better feature extraction step is performed, better will be the performance of classification.

- Ilan Steinberg et. al, proposed a method of recognizing hand gesture, by considering each pixel of the binary image as a feature. To reduce the complexity of computation for classification, the size of the images was reduced.[9]
- Omlin and Vapnik et. al., have recognized user hand gesture using color histogram as their feature, in which, the whole image was divided into smaller blocks and plot the corresponding histogram, for classification.[1]
- Mokhtar M. Hasan et. al., observed that the finger tip can also be used as a feature for recognition.[2]
- Nasser H. Dardas in, used a bag of feature for classifying hand gesture. In this approach, all the key points of the training images were extracted and these key points are mapped to its corresponding histogram feature vector for classification.[5]
- Oleg Rumyantsev et. al introduced an efficient method for recognizing hand gesture, based on PCA method, in which, the test image and database images were represented using Eigen values and the hand gestures were correctly classified using the Euclidean distance of these Eigen values.[6]

3.1 PREVIOUS SYSTEM

There are many systems that are proposed by different Authors given as following:

- The author Zhang Yuye System use AT89C51 and CAN BUS controller which leads to complicated design and cost of the system more because of CAN BUS controller. Also in this case power requirement will be more of AT89C51.[8]
- The author Manoj Kanta Mainali proposed a genetic algorithm approach to estimate the traffic volume in road sections without the traffic information of road sections. This method estimates the unknown traffic volume using only the known traffic volumes.[1]

- The author Cai Bai-gen designed a vehicle detection system based on magneto-resistive sensor is composed by wireless traffic information collection nodes which are set on two sides of road to detect vehicle signal. The magneto-resistive sensor is expensive and maintenance cost of the system will be more if the system fails.[4]

3.2 PROPOSED SYSTEM

To solve the problems, a system can be used called Hand Tracking And Gesture Recognition System. A primary goal of this gesture recognition is to create a system which can identify specific human gestures and use them to convey information for carrying out different task.[7]

Two methods are considered suitable for gesture recognition that are :-

- The first one is to use vision sensors like cameras to acquire images, which are analyzed to recognize the hand gestures.
- The second one is to place inertial sensor on the traffic police hand and extract the motion characters. The most advantage of the vision method is that it can recognize hand gestures without adding any extra hindrance to the system such as traffic controller. Also sensors are inconvenient for use while travelling.

So the purpose of this system is to control the traffic signals and mouse using hand gestures without using sensors at lower cost and with ease.

CHAPTER-04

HAND GESTURE

RECOGNITION

4.1 Introduction

Hand gesture recognition system is used for interfacing between computer and human using hand gesture. This project presents a technique for a human computer interface through hand gesture recognition that is able to recognize 26 static gestures from the American Sign Language hand alphabet. The objective of this project is to develop an algorithm for recognition of hand gestures with reasonable accuracy. [8]

Sign language recognition from hand motion or hand posture is an active area in gesture recognition research for Human Computer Interaction (HCI). A gesture is spatiotemporal pattern, which may be static or dynamic or both. Static morphs of the hands are called postures and hand movements are called gestures. The goal of gesture interpretation is to push the advanced human-machine communication to bring the performance of human-machine interaction close to human-human interaction. This is due to the existing complexities in hand tracking such as hand appearance, illumination variation, and inter-hands occlusion.[10]

The objective of this project is to develop a program that is able to detect gestures and track them in real time. It shall be done using simple signal processing algorithms LDA and PCA on images obtained from the webcam.

4.2 Hand Gesture Recognition

Gesture recognition pertains to recognizing meaningful expressions of motion by a human, involving the hands, arms, face, head, and/or body. It is of utmost importance in designing an intelligent and efficient human-computer interface. The applications of gesture recognition are manifold, ranging from sign language through medical rehabilitation to virtual reality. It is the process by which the gestures made by the user are recognized by the receiver. Gestures are expressive meaningful body motions involving physical movements of the fingers, hands, arms, head, face, or body with the intent of: 1) conveying meaningful information or 2) interacting with the environment. Gestures can be static (the user assumes a certain pose or configuration) or dynamic (with pre stroke, stroke, and post stroke phases). Some gestures also have both static and dynamic elements, as in sign languages.[3]



Figure 5: American Sign Language Hand Gestures

4.3 Types of Gestures

- **Hand and Arms Gestures** : recognition of hand poses, sign languages, and entertainment applications (allowing children to play and interact in virtual environments)
- **Head and Face Gestures** : some examples are: a) nodding or shaking of head; b) direction of eye gaze; c) raising the eyebrows; d) opening the mouth to speak; e) winking, f) flaring the nostrils; and g) looks of surprise, happiness, disgust, fear, anger, sadness, contempt, etc.[2]
- **Body Gestures**: involvement of full body motion, as in: a) tracking movements of two people interacting outdoors; b) analyzing movements of a dancer for generating matching music and graphics; and c) recognizing human gaits for medical rehabilitation and athletic training.[6]

They can also be classified as:

Gesticulation:- Spontaneous movements of the hands and arms that accompany speech.

Language-like gestures:- Gesticulation that is integrated into a spoken utterance, replacing a particular spoken word or phrase.

Pantomimes:- Gestures that depict objects or actions, with or without accompanying speech.

Emblems:- Familiar gestures such as V for victory, thumbs up, and assorted rude gestures.

Sign languages:- Linguistic systems, such as American Sign Language, which are well defined. [7]

4.4 Working Stages

Since the American Sign Language has a vast database with a complex system of facial expression and gestures for each word, initially work is done with individual letters.

- Stage 1- It deals with a primarily offline image based recognition system. The system is trained using images from an American Sign Language database training set and the efficiency of this classification will be tested using the testing database set.[15]
- Stage 2-Making the system to work with a input via a webcam. This tests the robustness of the system against different background, size and angle variation.[16]

4.5 Applications of Hand Gesture Recognition

Gesture recognition has wide-ranging applications such as the following:

- Developing aids for the hearing impaired
- Designing techniques for forensic identification
- Recognizing sign language
- Navigating and/or manipulating in virtual environments
- Communicating in video conferencing
- Distance learning/tele-teaching assistance
- Monitoring automobile drivers' alertness/drowsiness levels, etc.

Gesture based applications are broadly classified into two groups on the basis of their purpose: multidirectional control and a symbolic language.[14]

3D Design: CAD (computer aided design) is an HCI which provides a platform for interpretation and manipulation of 3-Dimensional inputs which can be the gestures.

Manipulating 3D inputs with a mouse is a time consuming task as the task involves a complicated process of decomposing a six degree freedom task into at least three sequential two degree tasks.[13]

Tele presence: There may raise the need of manual operations in some cases such as system failure or emergency hostile conditions or inaccessible remote areas. Often it is impossible for human operators to be physically present near the machines [4]. Tele presence is that area of technical intelligence which aims to provide physical operation support that maps the operator 4 arm to the robotic arm to carry out the necessary task.

Virtual reality: Virtual reality is applied to computer-simulated environments that can simulate physical presence in places in the real world, as well as in imaginary worlds. Most current virtual reality environments are primarily visual experiences, displayed either on a computer screen or through special stereoscopic displays [6]. There are also some simulations include additional sensory information, such as sound through speakers or headphones. Some advanced, haptic systems now include tactile information, generally known as force feedback, in medical and gaming applications.

Method	Application Area	Invariant factor
[19]	Real time system / control a computer graphic crane by hand gestures/ play games such as scissors/paper/stone	Lighting conditions / Translation
[14]	Sign Recognition	Lighting conditions / Translation
[8]	Sign Recognition	Rotation
[5]	Sign language	Rotation/ Translation/ Scaling
[16]	Robot control application	Translation/ Rotation / Scaling
[22]	Real-time system/ moderate computational resources devices e.g. netbooks	Rotation / Translation
[17]	Sign Recognition	Rotation/Translation/ Scaling
[18]	Sign Recognition	Rotation/Translation/ Scaling
[20]	Drawing graphical elements such as triangle, rectangular/ Editing graphical elements such as copy, paste, undo/ Mobile robot control/Virtual Reality.	-

TABLE 3: Application of Hand Gesture Recognition System

4.6 Challenges

There are many challenges associated with the accuracy and usefulness of gesture recognition software. For image-based gesture recognition there are limitations on the equipment used and image noise. Images or video may not be under consistent lighting, or in the same location. Items in the background or distinct features of the users may make recognition more difficult. [15]

The variety of implementations for image-based gesture recognition may also cause issue for viability of the technology to general usage. For example, an algorithm calibrated for one camera may not work for a different camera.

Method	# Recognized Gestures	# Total Gestures used For Training And Testing	Recognition Percentage	Database used
[8]	26	1040	DP 98.8%	American Sign Language (ASL)
			MLP 98.7%	
[5]	6	60	normal method 84%	Own Database
			Scaling normalization method 95%	
[31]	26	208	92.78%	American Sign Language (ASL)
[032]	0-9 numbers	298 video sequence for isolated gestures/ 270 video sequence for continuous gestures	90.45%	Recognize Arabic numbers from 0 to 9.
[20]	5 static/ 12 dynamic gestures	Totally 240 data are trained and then the trained are tested	98.3%	5 static gestures and 12 dynamic gestures.
[14]	31	130 for testing	90.45%	Own Database
[17]	6	60	100% for more than 4 gestures	Own Database
[18]	20	200	100% for 14 gestures, and >90 for 15-20 gestures	Own Database

TABLE 4: Testing Method For Hand Gesture Recognition System

CHAPTER-05

ALGORITHMS USED

5.1 Background Construction

Given the feed from the camera, the 1st thing to do is to remove the background. We use running average over a sequence of images to get the average image which will be the background too.

The running average image used for background subtraction So as and when we keep receiving image, we construct the new background average image as:

$$\text{CurBG}[i][j] = (\alpha)\text{CurBG}[i][j] + (1-(\alpha))\text{CurFrame}[i][j]$$

This equation works because of the assumption that the background is mostly static. Hence for those stationary item , those pixels arent affected by this weighted averaging Hence those pixels that are constantly changing isnt a part of the background, hence those pixels will get weighed down. Hence the stationary pixels or the background gets more and more prominent with every iteration while those moving gets weighed out. [3] Thus after a few iterations , you get the above average which contains only the background. In this case , even my face is part of the background as it needs to detect only my hands.



Figure 6: Running average Image Used For Background Subtraction

5.2 Background Subtraction

A simple method to start with is we can subtract the pixel values. However this will result in negative values and values greater than 255, which is the maximum value used to store an integer. Instead we use an inbuilt background subtractor based on a Gaussian Mixture-based Background/Foreground Segmentation Algorithm[1]. Background subtraction involves calculating a reference image, subtracting each new frame from this image and thresholding the result which results in a binary segmentation of the image which highlights regions of non-stationary objects. This reference image is the background image constructed. [14]

The changes given by the background subtraction is very light and is high in noise. To suppress the morphological transformations. Dilation operations consist of convoluting an image with some kernel (in our case is a 3x3 grid), which can have any shape or size, usually a square or circle. The kernel has a defined anchor point, usually being the center of the kernel. As the kernel is scanned over the image, we compute the maximal pixel value overlapped by and replace the image pixel in the anchor point position with that maximal value. As you can deduce, this maximizing operation causes bright regions within an image to grow (therefore the name dilation). Take as an example the image above. Applying dilation we can get: The background (bright) dilates around the black regions of the letter. Then we now perform erosion on it. This operation is the sister of dilation. What this does is to compute a local minimum over the area of the kernel. As the kernel is

scanned over the image, we compute the minimal pixel value overlapped by and replace the image pixel under the anchor point with that minimal value. Analogously to the example for dilation, we can apply the erosion operator to the original image (shown above). You can see in the result below that the bright areas of the image (the background, apparently),

get thinner, whereas the dark zones the writing gets bigger. Thus by performing this all the noise and a few spots inside a region are all cleaned there by we get a much clearer motion change image.

5.3 Contour Extraction

Contour extraction is performed using OpenCV's inbuilt edge extraction function. It uses a canny filter to get a list of contours. It works by convoluting a filter with the image so that gradually changing components get cancelled out while sudden changing components like at the edge or borders are enhanced. Thus the edges become visible. It does the following

steps to extract the edges.

- Filter out any noise. The Gaussian filter is used for this purpose.
- Apply a pair of convolution masks (in x and y directions)
- Find the gradient strength and direction. The direction is rounded to one of four possible angles (namely 0, 45, 90 or 135)
- Non-maximum suppression is applied. This removes pixels that are not considered to be part of an edge. Hence, only thin lines (candidate edges) will remain.
- Canny does use two thresholds (upper and lower):
 - If a pixel gradient is higher than the upper threshold, the pixel is accepted as an edge
 - If a pixel gradient value is below the lower threshold, then it is rejected.

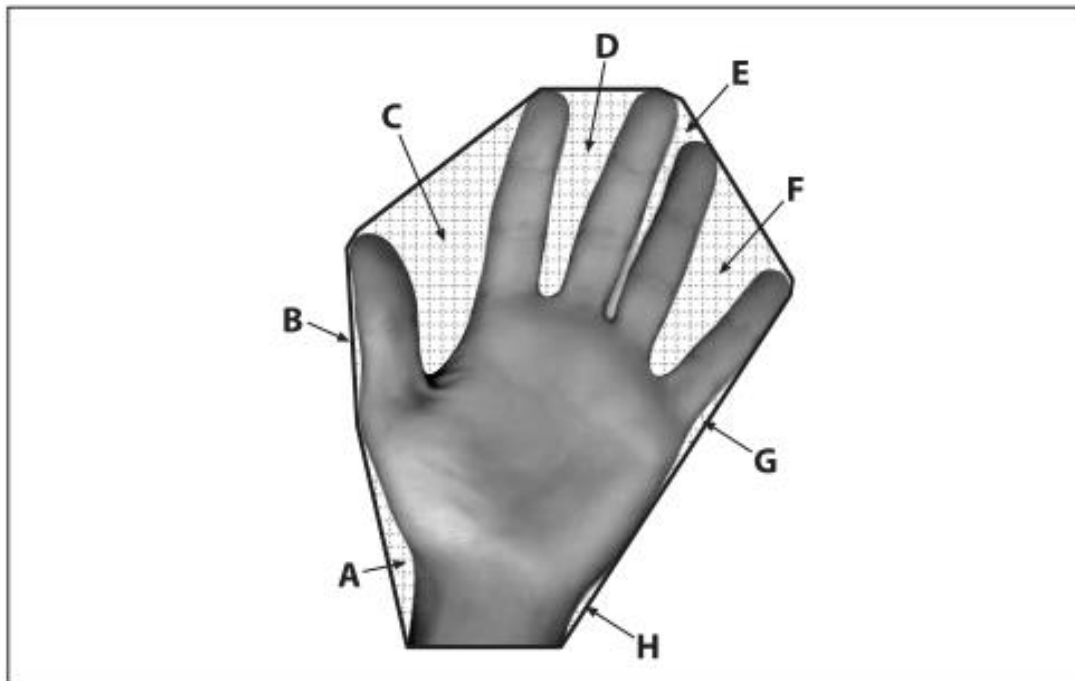


Figure 7: Contour Extraction

5.4 Convex Hull and Defects

Now given the set of points for the contour, we find the smallest area convex hull that covers the contours. The Sklanskys algorithm was used to find the convex hull which has a complexity of $O(n \log n)$. The observation here is that the convex hull points are most likely to be on the fingers as they are the extremities and hence this fact can be used to detect no of fingers. But since our entire arm is there, there will be other points of convexity too. So we find the convex defects ie, between each arm of the hull, we try to find the deepest point of deviation on the contour.

We find the smallest area convex hull that covers the contours. The observation here is that the convex hull points are most likely to be on the fingers as they are the extremities and hence this fact can be used to detect no of fingers. [12]

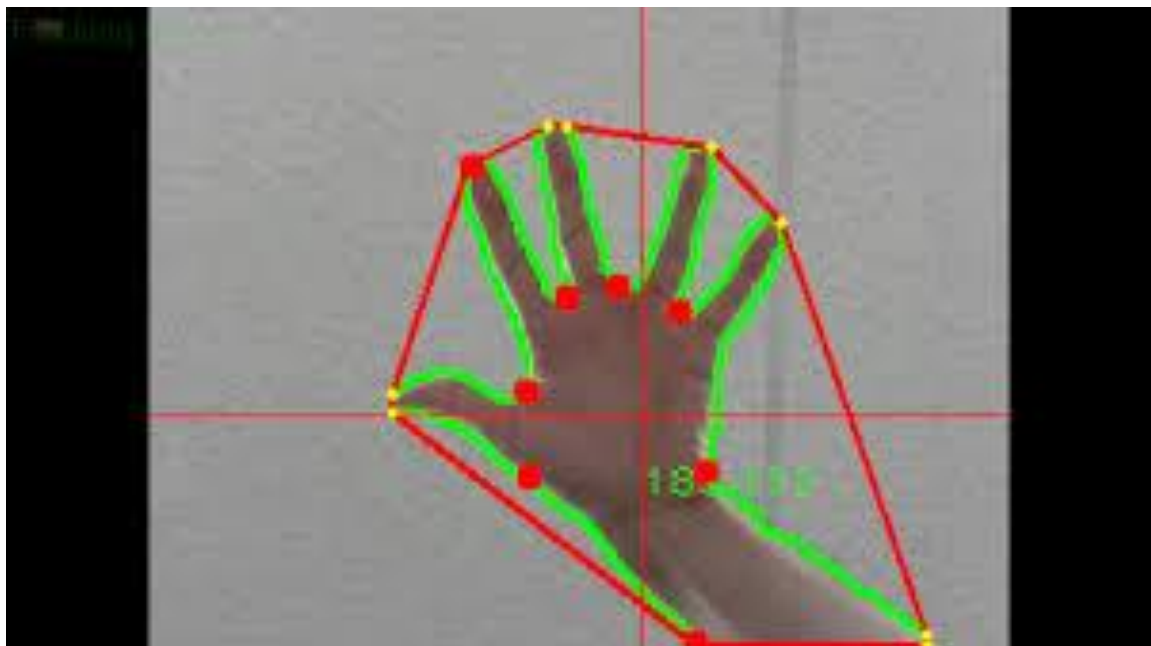


Figure 8: Contour Hull And Defects

5.5 Tracking and Finger Detection

Thus the defect points are most likely to be the center of the finger valleys as pointed out by the picture. Now we find the average of all these defects which is definitely bound to be in the center of the palm but its a very rough estimate. So we average out and find this rough palm center. Now we assume that the palm is angled in such a way that its roughly a circle.

So to find the palm center , we take 3 points that closes to the rough palm center and find the circle center and radius of the circle passing though these 3 points. Thus we get the center of the palm. Due to noise , this center keeps jumping, so to stabilize it , we take an average over a few iterations.[4]

Thus the radius of the palm is a indication of the depth of the palm and we know the center of the palm . Using this we can track the position of the palm in realtime and even know the depth of the palm using the radius. The next challenge is detecting the no of fingers.

We use a couple of observations to do this. For each maxima defect point which will be the finger tip, there will be 2 minimal defect points to indicate the valleys. Hence the maxima and the 2 minimal defects should form a triangle with the distance between the maxima and the minimas to be more or less same. Also the minima should be on or pretty close to the circumference of the palm. We use this factor too. Also the the ratio of the palm radius to the length of the finger trianger should be more or less same.

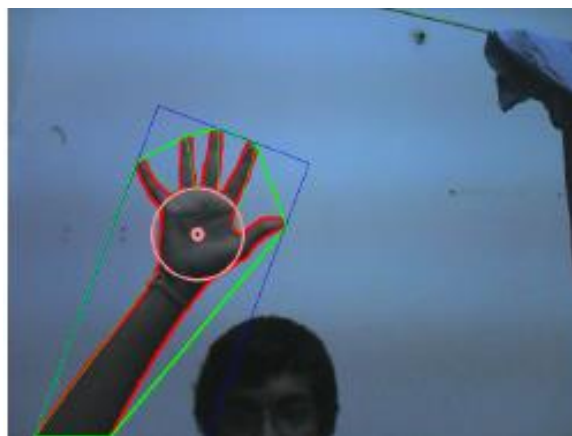


Figure 9:Finger And Palm Detection

CHAPTER-06

CONCLUSION

AND

FUTURE WORK

6.1 Conclusion

A program is created that was able to detect our hands, track them in realtime and perform some gesture recognition with simple signal processing performed on images obtained from a regular laptop web-camera. Reasonable accuracy and stability was obtained which can be used for steady and simple gestures to perform tasks.

The results also showed that the gesture recognition application was quite robust for static images. Sometimes the adjustment was difficult to do because of the lighting conditions and the amount of objects in the background. [10] The application was very susceptible to noise on the video stream. Slight hand movements could affect gesture recognition. Nevertheless, if the hand is steady enough for long enough, the program outputs the correct command. It was also observed that while the program was executing there were memory leaks. Attempts to remedy the problem were made by using the OpenCV functions to release memory. Despite this, the leaks continued. Perhaps the leaks were due to the implementation of OpenCV functions for the sequences behind the scenes.

For integrating the program with the robot in the future, it would be necessary to consider other output such as speed or velocity as part of the navigational control commands. Based on the results, a computer vision application could detect and recognize simple hand gestures. While the use of moment invariants was not considered suitable because the same gestures could be used pointing in opposite directions, other learning algorithms like Background Detection, subtraction, Contour extraction ,etc could be explored to make the program more robust and less affected by extraneous objects and noise.

6.2 Future Work

We can expand on the system in the future to improve detection and tracking to overcome the limitations, for example apart from using just skin colour for detection , we can use some other properties , similarly we can use other techniques for tracking and especially we can work on making the recognition phase more autonomous and recognize the gestures in real time as well. [1]

While the system works fairly well for three different hand gesture recognition, there is still room for improvement. Even though we have designed the system for varying illumination the skin detection step still misclassifies some pixels. One improvement to the current system would be gesture recognition based on template matching. On doing so we can differentiate between multiple single-finger gestures. Left/right hand detection could be taken into account for additional gesture recognition.

The current system performs a number of image processing including filtering and segmentation on every acquired image. Even though the speed of our system in gesture recognition is quite good, we could possibly improve the recognition rate significantly by tracking local features like hands and fingers. We have already used Kalman filters to track the hand position. But this tracking is limited to tracking position only. As pointed out in paper [8] this tracking can be transferred to the entire segmented hand region. The only downfall in this approach would be that the hand must not move too quickly. But in such a case reinitialization of tracker based on current implementation could be performed.

6.3 SCREENSHOTS

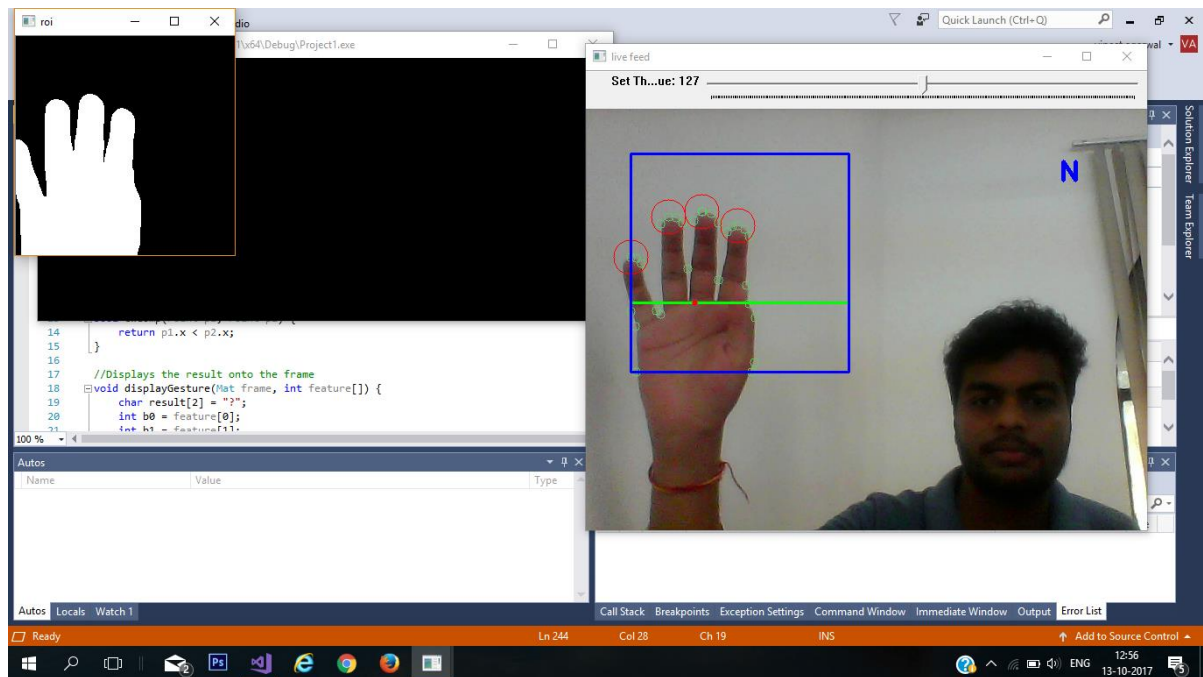


Figure 10: Screenshot-1

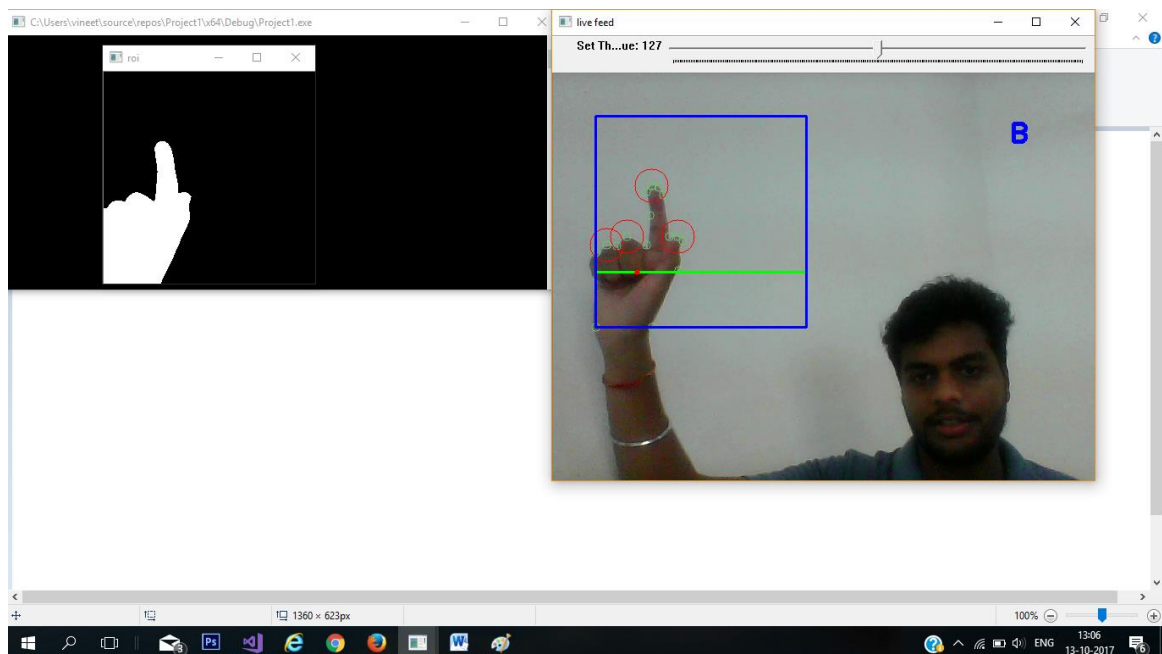


Figure 11: Screenshot-2

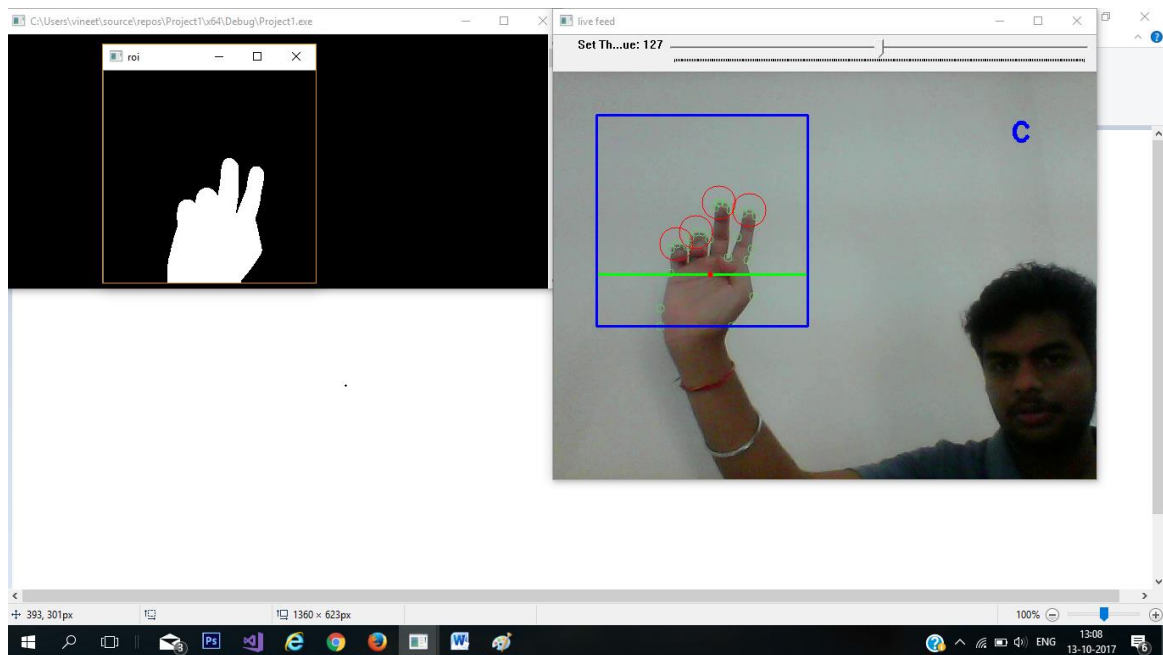


Figure 11: Screenshot-3

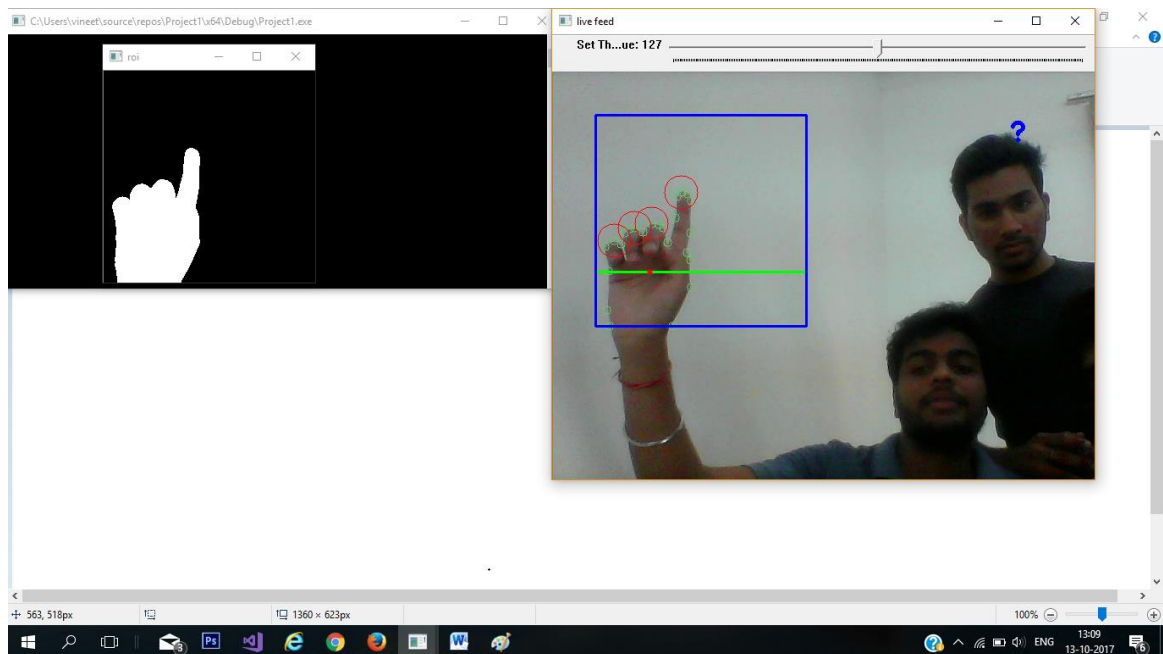


Figure 12: Screenshot-4

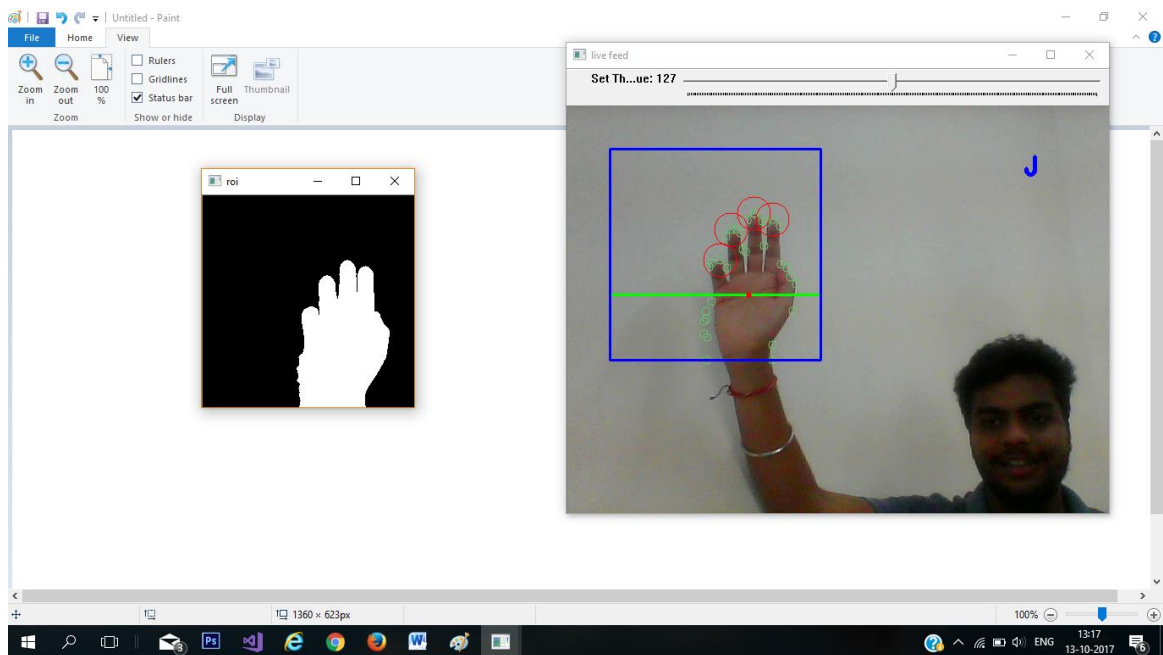


Figure 13: Screenshot-5

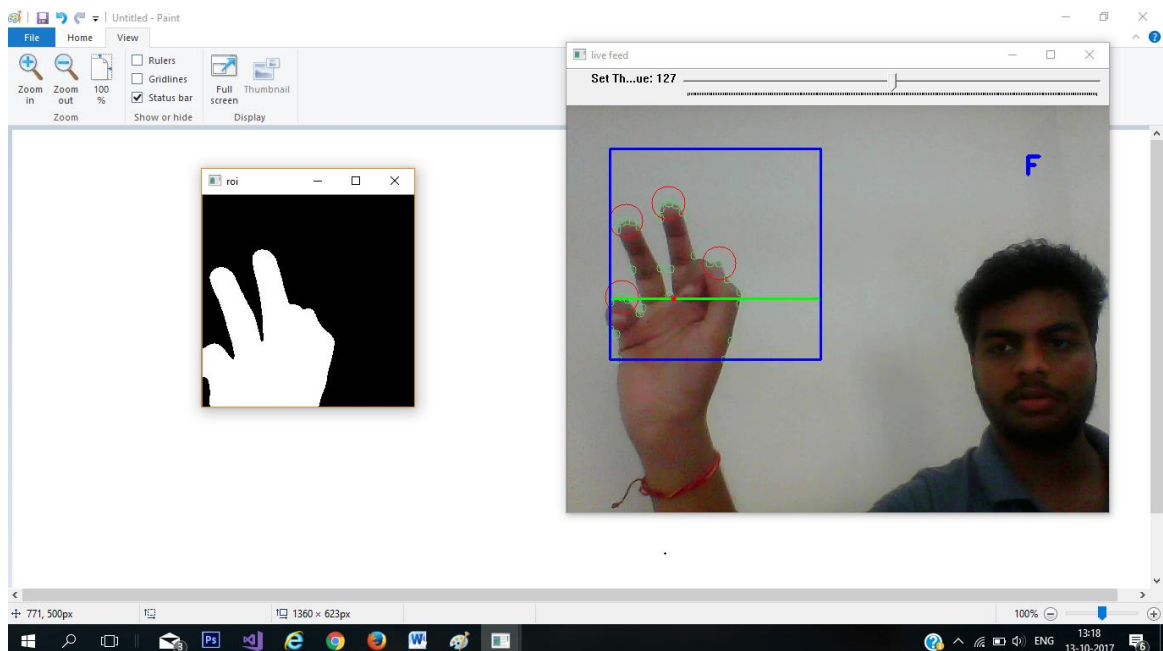


Figure 14: Screenshot-6

CHAPTER-07

REFERENCES

- [1] A. Pradhan, M. Ghose, and M. Pradhan, "A hand gesture recognition using feature extraction," *Int J Curr Eng* ..., pp. 323–327, 2012.
- [2] M. Abastillas, "Real-Time Hand Gesture Detection and Recognition Using Simple Heuristic Rules."
- [3] A. Dhote and S. C. Badwaik, "Hand tracking and gesture recognition," *2015 International Conference on Pervasive Computing (ICPC)*. pp. 1–5, 2015.
- [4] M.K. Hu, "Visual pattern recognition by moment invariants," *IRE Transactions on Information Theory*, vol. 8, pp. 179-187, 1962.
- [5] I. Abstract and X. P. Description, "CS4670 Final Report Hand Gesture Detection and Recognition for Human-Computer Interaction."
- [6] S. K. Chen, "Automatic Hand Gesture Recognition," no. April, pp. 1–55, 2015.
- [7] "Hand Detection and Gesture Recognition using ASL Gestures Supervisor : Andre L . C . Barczak Student : Dakuan CUI," *Training*.
- [8] D. Gurung, C. Jiang, and J. Deray, "Hand Gestures Recognition and Tracking To cite this version : University of Burgundy Hand Gesture Recognition and Tracking," 2013.
- [9] S. D. Badgujar, G. Talukdar, and O. Gondhalekar, "Hand Gesture Recognition System," *Int. J. Sci. Res. Publ.*, vol. 4, no. 2, pp. 1–5, 2014.
- [10] Q. M. S. Tier, Q. M. S. P. Subject, Q. Objectives, P. Doc, R. By, and K. J. Page, "Quality Objectives and Planning Procedure," no. November, 2014.
- [11] R. Zaman Khan, "Hand Gesture Recognition: A Literature Review," *Int. J. Artif. Intell. Appl.*, vol. 3, no. 4, pp. 161–174, 2012.
- [12] M. Tang. "Hand Gesture Recognition Using Microsoft's Kinect." Paper written for CS229, March 16, 2011.
- [13] Jayashree R. Pansare¹, Shravan H. Gawande², Maya Ingle³ *Journal of Signal and Information Processing*, 2012, 3, 364-367 doi:10.4236/jsip.2012.33047 Published Online August 2012

- [14] Rachel Smith, Senior member(IEEE), “A Fast Algorithm for Vision-Based Hand Gesture Recognition for Robot control.”
<http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1659822>
- [15] J.R. Parker, Algorithms for Image Processing & Computer Vision, Wiley Publishing, 2011
- [16] Vaughn M. Segers, “Real-Time Gesture Recognition using Eigenvectors,” Department of Computer Science, University of the Western Cape, vol 4, pp- 7348,1999
- [17] Sushmita Mitra, Senior Member(IEEE) and Tinku Acharya, Senior Member(IEEE) “Gesture Recognition : A Survey”, VOL. 37, NO. 3, MAY 2007 311
- [18]Color Based Hand and Finger Detection Technology for User Interaction
<http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4622829>
- [19]Christopher William, “Tutorial on Fisher Linear Discriminant Analysis”
http://www.ics.uci.edu/~welling/classnotes/papers_class/Fisher-LDA.pdf
- [20] Min B., Yoon, H., Soh, J., Yangc, Y., & Ejima, T. (1997). “Hand Gesture Recognition Using Hidden Markov Models”. IEEE International Conference on computational cybernetics and simulation. Vol.5, Doi: 10.1109/ICSMC.1997.637364
- [21] Verma, R., Dev A. (2009).”Vision based hand gesture recognition using finite state machines and fuzzy logic”. IEEE International Conference on Ultra-Modern Telecommunications & Workshops (ICUMT '09), pp. 1-6. doi: 10.1109/ICUMT.2009.5345425
- [22] Luigi Lamberti, Francesco Camastra, (2011). “Real-Time Hand Gesture Recognition Using a Color Glove”, Springer Proceedings of the 16th international conference on Image analysis and processing: Part I ICIAP.

- [23] Minghai Y., Xinyu Q., Qinlong G., Taotao R., Zhongwang L., (2010). "Online PCA with Adaptive Subspace Method for Real-Time Hand Gesture Learning and Recognition", journal World Scientific and Engineering Academy and Society WSEAN, Vol. 9(6).
- [24] N. A. Ibraheem., R. Z. Khan, (2012). "Vision Based Gesture Recognition Using Neural Networks Approaches: A Review", International Journal of Human Computer Interaction (IJHCI), Malaysia, Vol. 3(1).
- [25] J.R. Parker, Algorithms for Image Processing and Computer Vision, Wiley Publishing, 2011
- [26] R. Mukundan and K.R. Ramakrishnan, Moment Functions in Image Analysis, World Scientific, 1998.
- [27] M.K. Hu, "Visual pattern recognition by moment invariants," IRE Transactions on Information Theory, vol. 8, pp. 179-187, 1962.
- [28] Y. Freund and R.E. Schapire, "A short introduction to boosting," Journal of Japanese Society