Sudan University of Science and Technology
College of Graduate Studies

A Hand Gesture Recognition System for Deaf-Mute Individuals

نظام للتعرف على إيماءة اليد للأفراد ذوي الاعاقة السمعية الكلامية

A thesis Submitted in Partial Fulfillment of the Requirement for the Degree of M.Sc. in Mechatronics Engineering

Prepared By:
Rugia Said Thabit KamalEldeen

Supervisor:
Dr. Ebtihal Haider Gismalla Yousif

March 2019

الإستهلال

بِسْمِ اَللَّهِ اَلرَّحْمٰنِ اَلرَّحِيمِ

اَللَّهُ نورُ اَلسَّمٰوٰتِ وَالْأَرْضِ مَثَلُ نورِهِ كَمِشْكٰوةٍ فيها مِصْباحٌ اَلْمِصْباحُ فى زُجاجَةٍ اَلزُّجاجَةُ كَأَنَّها كَوْكَبٌ دُرِّىٌّ يوقَدُ مِنْ شَجَرَةٍ مُبٰرَكَةٍ زَيْتونَةٍ لا شَرْقِيَّةٍ وَلا غَرْبِيَّةٍ يَكادُ زَيْتُها يُضيءُ وَلَوْ لَمْ تَمْسَسْهُ نارٌ نورٌ عَلىٰ نورٍ يَهْدِى اَللَّهُ لِنورِهِ مَنْ يَشاءُ وَيَضْرِبُ اَللَّهُ اَلْأَمْثٰلَ لِلنّاسِ وَاللَّهُ بِكُلِّ شَيْءٍ عَلِيمٌ ﴿٣٥﴾

سورة اَلنورْ

Dedication

To all those whose massive support made me achieve new heights in my life.

My Family and Friends

For all those and more, I present my humble thesis.

Acknowledgments

I want to thank Almighty ALLAH first, the creator and the owner of this universe, who provided me guidance, strength and ability to complete this research. I would like to express my deepest appreciation to my parent, my husband, my sister, my daughter and my little baby from the bottom of my heart for their help, guidance, patient and support in completion of this project. I am very thankful to Dr.Ebtihal Haider Gismalla Yousif my thesis supervisor for all her assistance during this master thesis work

Abstract

A deaf-dumb individual always uses gestures to convey his/her ideas to others. However, it is always difficult for a normal person to understand this gesture language. The basic objective of this project is to develop a computer based system to recognize 26 gestures from American Sign Language (ASL) using MATLAB, which will enable deaf-dumb individuals significantly to communicate with all other people using their natural hand gestures. The proposed system in this thesis is composed of four modules, which are preprocessing and hand segmentation, feature extraction, sign recognition and text of sign voice conversion. Segmentation is done by converting image to Hue-Saturation-Value (HSV) formate and using color threshold APP. Blob features are extracted by using Bag of feature that is used the Speed Up Robust Features (SURF) algorithm. Furthermore, the K-Nearest Neighbor (KNN) and Support Vector Machine (SVM) algorithms are used for gesture recognition. Finally, the Recognized gesture is converted into voice format. To make the system more user friendly, a Graphical User Interface (GUI) is designed. To train and test the proposed model, a self-collected dataset for ASL was prepared using hand gestures from both male and female volunteers, who have different ages and skin color in different background and postures by an ordinary phone camera. The implemented model demonstrates that the average accuracy from evaluation set is (89%), whereas the average accuracy obtained from test set is (84.6%).

المستخلص

يستخدم الاشخاص ذوي الاعاقه السمعية والبصرية الأشارات في إيصال أفكارهم إلى الآخرين وعادة ما يواجه الأشخاص السليمين صعوبات في فهم لغة الأشارات، لذلك كان الهدف الأساسي لهذا المشروع هو تطوير نظام للتعرف على ست و عشرين أشارة من لغة الإشارة الامريكية و ذلك لتمكين الأشخاص ذوي الإعاقة السمعية والبصرية من التواصل بسهولة مع الآخرين باستخدام الأشارات الاعتيادية لأيديهم. النظام المقترح في هذا البحث يتكون من أربعة أجزاء و هي المعالجة ألاولية للصور وفصل صورة اليد من الخلفية وأبرازها و أستخراج خصائص الصورة و مطابقة الصوره مع الحرف وتحويل النص إلى صوت. فصل صورة اليد يتم بتحويل الصوره إلي الصيغة اللونية صشق ومن ثم أستخدام بريجة عتبة اللون. الخصائص الفقاعية أستخرجت من الصورة بأستخدام نموذج حقيبةالخصائص والتي تستخدم خوارزمية تسريع الميزات القوية. كذلك لمطابقة الصوره مع نص الحرف أستخدمت خوارزمية اقرب عدد من الجيران وخوارزمية متجه الدعم الآلي ومن ثم لتحويل النص إلى صوت. كذلك لجعل النظام أكثر سهولة في ألاستخدام تم تصميم واجهة للمستخدم. لتدريب وأختبار النظام أستخدمت بيانات تم تجميعها ذاتيا من متطوعين ذكور و إناث بمختلف الاعمار ولديهم ألوان بشرة مختلفة في خلفيات ووضعيات مختلفه بأستخدام كاميراالهاتف الذكي. النتائج المتحصل عليها تشير بأن متوسط دقة تقييم النظام ٨٩% ومتوسط دقة أختبار النظام ٨٤.٦%

Table of Contents

List of Figures

List of Tables

List of Abbreviations

ASL	American Sign Language
BOF	Bag of Feature
BOW	Back of Word
CRT	Cathode Ray Tube
DOG	Difference-of-Gaussians
ECOC	error-correcting output codes
GLCM	Gray Level Co-Occurrence Matrix
GSM	Global System for Mobile
GUI	Graphical User Interface
HMM	Hidden Markov Model
HSI	hue, saturation and luminance
HSL	hue, saturation and intensity
HSV	hue, saturation and value
KNN	K-Nearest Neighbor
MLP	Multi-Layer Perceptron
MSER	Maximally Stable Extremal Regions
PCA	Principal Component Analysis
RBF	Radial Basis Function
RGB	Red, Green and Blue
SIFT	Scale Invariant Feature Transform
SURF	Speed Up Robust Features
SVM	Support Vector Machine
TTS	text-to-speech
TV	Television

List of Symbols

H	hue
S	saturation
V	Value
R	Red
G	Green
B	Blue
m_k	Nearest Cluster Center
x_i	Any Point
w	the normal vector to planes
b	determines the location relative to the origin
A	Label of Class

Chapter one
Introduction

1.1 Introduction

Communication between normal and handicapped person such as deaf people, dumb people, and blind people has always been a challenging task. It has been observed that they find it really difficult at times to interact with normal people with their gestures, as only a very few of those are recognized by most people. Since people with hearing impairment or deaf people cannot talk like normal people so they have to depend on some sort of visual communication in most of the time. Sign Language is the primary means of communication in the deaf and dumb community. As like any other language it has also got grammar and vocabulary but uses visual modality for exchanging information [1]. The hearing impaired people becomes neglected from the society because the normal people never try to learn ASL nor try to interact with the hearing impaired people. This becomes a curse for them and so they mostly remain uneducated and isolated. So the only way for enhance the communication between mute people and normal people is recognition of sign language and converting it to the corresponding voice [2]. Sign language recognition was developed in the '90s. Research related to hand gesture can be classified into two parts. In the first part, electromagnetic gloves and sensors are introduced which consist the hand shape, movements and orientation of the hand. These have limitation such as cost and not suitable for practical use. Second one is computer vision based gesture recognition system, consist of image processing techniques. Which is require only a camera, and computer or mobile device like phones or tablets which are very common among people of all ages They are connected with network and provide seamless communications through internet or cellular services thus realizing a natural interaction between humans and computers without the use of any extra devices [3].

1.2 Problem Statement

One of the important problems that our society faces is that people with disabilities are finding it hard to come up with the fast growing technology. In the recent years, there has been a rapid increase in the number of hearing impaired and speech disabled victims due to birth defects, oral diseases and accidents. When a deaf-dumb person speaks to a normal person, the normal person seldom understands and asks the deaf-dumb person to show gestures for his/her needs.

1.3 Proposed Solution

In order to bridge the gap in communication among deaf and dumb community and normal community, manual interpreter translates the hand signs to voice to help communication at both side, this research proposed a system using an image processing that will be useful for deaf-mute individual.

1.4 Aims and Objective

The overall purpose of this project is facilitate the interaction between deaf-dumb people and normal people to makes the communication between normal people and dumb people easier, by translate the sign language to voice or text with high accuracy.

1.5 Methodology

After literature review, to accomplish the required objectives from this project, there are three steps:

- First step: Collect the data by taking images of gesture.
- Second step: Processing the image using Matlab.
- Third step: Skin color detection.
- Fourth step: Extract Features from images.
- Fifth step: cluster the features.
- Sixth step: classify the features.
- seventh step: Training the data.
- eighth step: Testing the system.

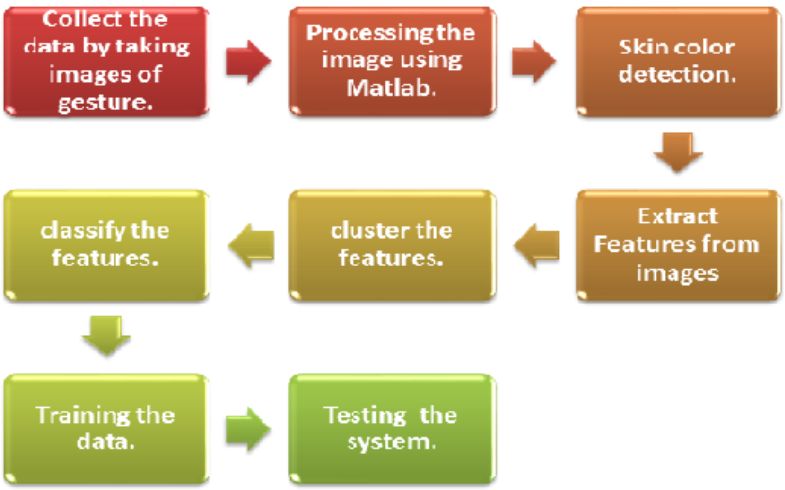

Figure 1.1: The Methdology

1.6 Thesis Organization

The rest of this thesis will be organized as follows. Chapter two is composed of introduction to American sign language and discusses relevant research. Chapter three discusses the algorithmic approach used in this research and the steps of implementation. Chapter four discusses and analyzes the final outcomes. Finally, Chapter five presents the conclusions and recommendations.

Chapter Two
Background and Literature Review

2.1 Introduction

Dumb people are usually deprived of normal communication with other people in the society. It has been observed that they find it really difficult at times to interact with normal people with their gestures, as only a very few of those are recognized by most people. Since people with hearing impairment or deaf people cannot talk like normal people so they have to depend on some sort of visual communication in most of the time. Sign Language is the primary means of communication in the deaf and dumb community. As like any other language it has also got grammar and vocabulary but uses visual modality for exchanging information. The problem arises when dumb or deaf people try to express themselves to other people with the help of these sign language grammars. This is because normal people are usually unaware of these grammars. As a result it has been seen that communication of a dumb person are only limited within his/her family or the deaf community [4].

2.2 Sign Language

Sign Language is the means of communication among the deaf and mute community. Sign Language emerges and evolves naturally within hearing impaired community. Sign Language communication involves manual and non-manual signals where manual signs involve fingers, hands, arms and non-manual signs involve face, head, eyes and body. Sign Language is a well-structured language with a phonology, morphology, syntax and grammar. Sign language is a complete natural language that uses different ways of expression for communication in everyday life. Sign language is mostly used by the deaf, dumb or people with any other kind of disabilities [5].

2.2.1 History of Sign language

In the western society sign language started developing in 17th century for a visual language. It is a language with the combination of conventional gesture, hand signs, finger spelling also includes the position of the hand positions to represent a meaningful line. The first American school for the deaf was founded in 1817 by Laurent Clerc and Thomas Hopkins Gallaudet. They are said to be the first founder of American Sign Language. This is actually partly true. Laurent Clerc was from Europe and taught French Sign Language. Thomas Hopkins Gallaudet brought Clerc back to America to start the first American school for the deaf. Like Abbe Charles Michel de L'Epee's school, children from all over the country traveled to attend this school, bringing their home-signs with them. These home-signs, combined with French Sign Language, became American Sign Language. Before 19th century sign languages were confined only to fixed words using finger spelling system. It evolved gradually and now many researches are going on for interpreting real time sign languages to make it understanding easier for everyone. Some inventors acclaim mankind as the originators of the first sign language. This is probably true. Early man, before spoken language, probably used gesture [6].

2.2.2 American Sign Language

A gesture may be defined as a movement, usually of hand or face that expresses an idea, sentiment or emotion e.g. rising of eyebrows, shrugging of shoulders is some of the gestures we use in our day life. Sign language is a more organized and defined way of communication in which every word or alphabet is assigned some gesture.

American Sign Language (ASL) is a complete, complex language that employs signs made by moving the hands combined with facial expressions and postures of the body. It is the primary language of many North Americans who are deaf and is one of several communication options used by people who are deaf or hard-of-hearing.

ASL is a language completely separate and distinct from English. It contains all the fundamental features of language—it has its own rules for pronunciation, word order, and complex grammar. While every language has ways of signaling different functions, such as asking a question rather than making a

statement, languages differ in how this is done. For example, English speakers ask a question by raising the pitch of their voice; ASL users ask a question by raising their eyebrows, widening their eyes, and tilting their bodies forward. Just as with other languages, specific ways of expressing ideas in ASL vary as much as ASL users do. In addition to individual differences in expression, ASL has regional accents and dialects. Just as certain English words are spoken differently in different parts of the country, ASL has regional variations in the rhythm of signing, form, and pronunciation. Ethnicity and age are a few more factors that affect ASL usage and contribute to its variety [7].

In American Sign Language, each alphabet of English vocabulary, A-Z, is assigned a unique gesture as shown in Figure 2.1.

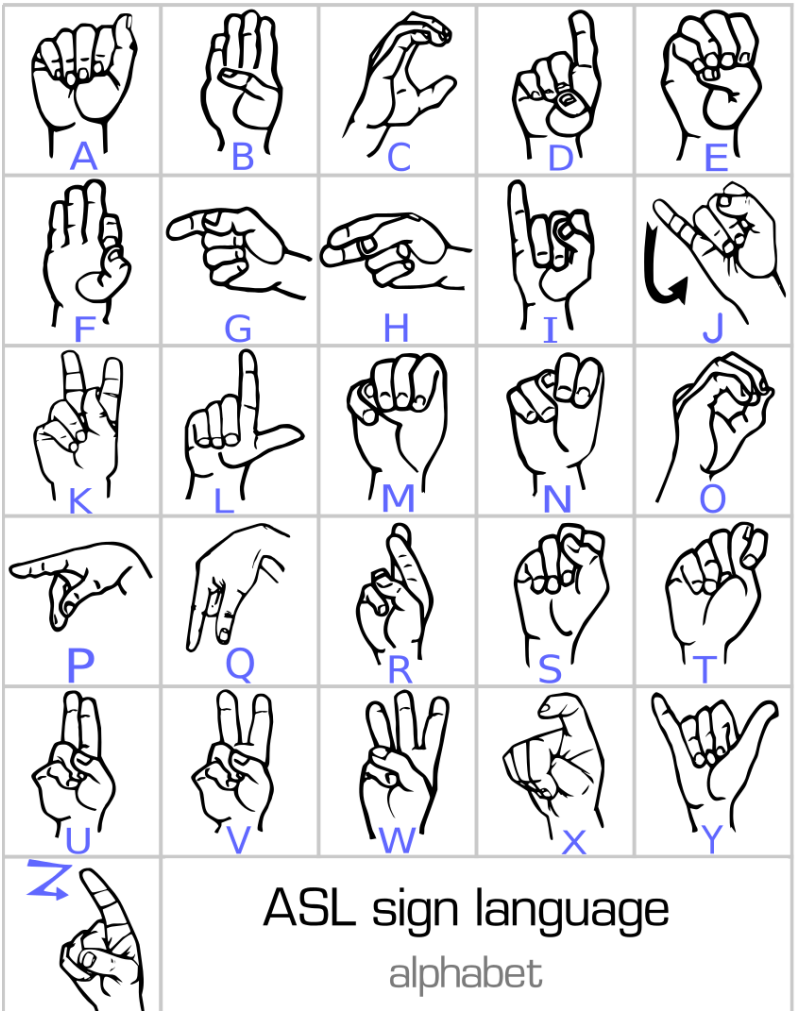

Figure 2.1: American Sign Language Alphabets

2.3 Sign Language Recognition System

Sign Language recognition system transfers the communication from human-human to human-computer interaction. The aim of the sign language recognition system is to present an efficient and accurate mechanism to transcribe text or speech, thus the "dialog communication" between the deaf and hearing person will be smooth. There is no standardized sign language for all deaf people across the world. However, sign languages are not universal, as with spoken languages, these differ from region to region. A person who can talk and hear properly (normal person) cannot communicate with deaf and dumb person unless he/she is familiar with sign language. Same case is applicable when a deaf and dumb person wants to communicate with a normal person or blind person. There are two types of sign language recognition.

1. Sensor Based Approach: In this approach for hand gesture recognition different types of sensors were used and placed on hand, when the hand performs any gesture, the data is recorded and is then further analyzed. Sensor based approach damages the natural motion of hand because of use of external hardware. The major disadvantage is complex gestures cannot be performed using this method, very expensive and brings much cumbersome experience to the users.

2. Vision Based Approach: In this approach camera takes the image of gesture, extracts the main feature and recognizes it , the main disadvantage is person and camera independent to achieve real time performance. But because of few requirements they are considered easy, natural and less costly compared to glove based approach [8].

2.4 Relevant Background on Image Processing

2.4.1 Color Space Types

There are three types of color spaces, explained as follows.

1. *RGB*, which is most common color space for image representation. It was developed with CRT. RGB color space contains three additive primary colors; red, green and blue. These colors combine in different ratios additively to produce other colors.

2. The second type is HSV, HSI, HSL (hue, saturation, value/intensity/-luminance). Hue describes the central color in pixel for example pink, green, purple etc. Saturation means the thickness of color i.e. measures the colorfulness. Hue and saturation taken together are called chromaticity and I or V is the intensity value associated with brightness.
3. The third type is orthogonal color spaces (YCbCr,YIQ,YUV,YES). These color spaces isolates RGB channels into chrominacity and luminancity information. YCbCr color space was defined in response to increasing demands for digital algorithms to handle video information, and is commonly used by European TV system [9].

In table 2.1 comparison of most three popular color spaces is presented.

Table 2.1: Comparison of The Most Three Popular Color Spaces

Color Space	Advantages	Disadvantages
RGB	1. Simplest method 2. suitable for video display	1.Sensitive to illumination intensity. 2. Device Dependent
HSV	1. Good with histogram operations 2. Separates the chromatic part from achromatic part 3. Easy human perception	1. perceptual relevance for computation speed 2. User oriented -Device dependent 3. sensitivity to value RGB value deviation 4. unstable hue, because of the angles
Ycbcr	1. Separated chrominance and luminance 2. Good for image compression	1. Device Dependent 2. Depends on the RGB primaries

2.4.2 Skin Detection

Skin detection means detecting those pixels and regions from an image that contain human skin tone color in an image. Skin color detection has extensive applications in different disciplines of life such as entertainment world, advertisement, medical system, porn image filtering, defense systems,robotics and in various other industries. Use of color information as a feature for skin detection enables fast processing and brings robustness to such application. Skin detection methods can be further classified as follows.

1. The *region based method*, which takes the spatial arrangement of neighboring pixels to improve performance of skin pixel detection.
2. The *pixel based method*, where each pixel is categorized in form of skin pixel and non-skin pixel apart from neighboring pixels, Pixel based methods are fast in processing and more appropriate for different geometric patterns in skin color. Moreover as they are more sensitive to resolution changes hence they don't need the use of artificial color signs in image [9].

2.5 Bag of Feature

The past decade has seen the rise of the Bag of Features approach in computer vision. Bag of Features (BoF) methods have been applied to image classification, object detection, image retrieval and even visual localization for robots. BoF approaches are characterized by the use of an order less collection of image features. Lacking any structure or spa-tail information, it is perhaps surprising that this choice of image representation would be powerful enough to match or exceed state-of-the-art performance in many of the applications to which it has been applied. Due to its simplicity and performance, the Bag of Features approach has become well-established in the field. A Bag of Features method is one that represents images as order less collections of local features. The name comes from the Bag of Words representation used in textual information retrieval.

To explain BOF represent a document as a normalized histogram of word counts, commonly, one counts all the words from a dictionary that appear in the document. This dictionary may exclude certain non-informative words such as articles (like "the"), and it may have a single term to represent a set of synonyms. The term vector that represents the document is a sparse vector where each element is a term in the dictionary and the value of that element is the number of times the term appears in the document divided by the total number of dictionary words in the document (and thus, it is also a normalized histogram over the terms). The term vector is the Bag of Words document representation – called a "bag" because all ordering of the words in the document have been lost. The Bag of Features image representation is analogous. A visual vocabulary is constructed to represent the dictionary by clustering features extracted from a set of training images. The image

features represent local areas of the image, just as words are local features of a document. Clustering is required so that a discrete vocabulary can be generated from millions (or billions) of local features sampled from the training data. Each feature cluster is a visual word. Given a novel image, features are detected and assigned to their nearest matching terms (cluster centers) from the visual vocabulary. The term vector is then simply the normalized histogram of the quantized features detected in the image. The steps of bag of feature shown in Figure 2.2 [10].

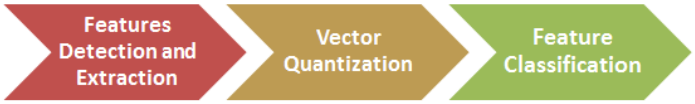

Figure 2.2: Steps Of Bag Of Feature

2.5.1 Feature Detection and Descriptor

Feature detection is the process of computing the abstraction of the image information and making a local decision at every image point to see if there is an image feature of the given type existing in that point. Feature detection and image matching have been two important problems in machine vision and robotics, and their applications continue to grow in various fields. An ideal feature detection technique should be robust to image transformations such as rotation, scale, illumination, noise and affine transformations. In addition, ideal features must be highly distinctive, such that a single feature to be correctly matched with high probability [11].

2.5.1.1 Feature Detection

Feature detection is the process of deciding where and at what scale to sample an image. The output of feature detection is a set of keypoints that specify locations in the image with corresponding scales and orientations. These keypoints are distinct from feature descriptors, which encode information from the pixels in the neighborhood of the keypoints type of feature detection

1. Interest Point Operators:
   While there are many variations, an interest point operator typically detects keypoints using scale space representations of images. A scale space represents the image at multiple resolutions, and is generated

by convolving the image with a set of guassian kernels. Interest point operators detect locally discriminating features, such as corners, blob-like regions, or curves. Example of key points detection is a Difference-of-Gaussians (DoG). The Maximally Stable Extremal Regions (MSER), the Harris-Affine.

2. Visual Saliency.
3. Deterministic a Random Sampling [10].

2.5.1.2 Feature Descriptors

determine how to represent the neighborhood of pixels near a localized region,The simplest approach is to simply use the pixel intensity values, scaled for the size of the region, or an eigenspace representation thereof. The most popular feature descriptor in the BoF literature is the SIFT (Scale Invariant Feature Transform) descriptor An alternative to the SIFT descriptor that has gained increasing popularity is SURF (Speeded Up Robust Features) (Bay et al, 2006). The SURF algorithm consists of both feature detection and representation aspects. It is designed to produce features akin to those produced by a SIFT descriptor on Hessian-Laplace interest points, but using efficient approximations. Reported results indicate that SURF provides a significant speed-up while matching or improving performance [10].

2.5.2 Quantization And Distance Measures (Clustering)

Vector Quantization (Clustering) is used to build the visual vocabulary in Bag of Features algorithms. There are a great many clustering/vector quantization algorithms, Many BoF implementations are described as using K-means.Given any clustering method, there will be points that are equally close to more than one centroid. These points lie near a Voronoi boundary between clusters and create ambiguity when assigning features to terms. With K-means and similar clustering methods, the choice of initial centroid positions affects the resultant vocabulary. When dealing with relatively small vocabularies, one can run K-means multiple times and select the best performing vocabulary during a validation step. This becomes impractical for very large data sets. When determining the distance between two features, as required by clustering and term assignment, common choices are the Manhattan (L1), Euclidean (L2), or Mahalanobis distances. A distance measure is also needed in term vector space

for measuring the similarity between two images for classification or retrieval applications Euclidean and Manhattan distances over sparse term vectors can be computed efficiently using inverted indexes and are thus popular choices. However, the relative importance of some visual words leads to the desire to weight term vectors during the distance computation. The following subsections provide more details on these and other issues.

2.5.2.1 Term Weights

One of the earliest strategies for handling quantization issues at a gross level is to assign weights to the terms in the term vector. This can be viewed as a mitigation strategy for quantization issues that occur when the descriptors are distributed in such a way that simple clustering mechanisms over-represent some descriptors and under represent others.

2.5.2.2 Soft Assignment

A given feature may be nearly the same distance from two cluster centers, but with a typical "hard assignment" method, only the slightly nearer neighbor is selected to represent that feature in the term vector. Thus, the ambiguous features that lie near Voronoi boundaries are not well-represented by the visual vocabulary. To address this problem, researchers have explored multiple assignments and soft weighting strategies.Multiple assignment is where a single feature is matched to k nearest terms in the vocabulary. Soft weights are similar, but the k nearest terms are multiplied by a scaling function such that the nearest term gets more weight than the k'th nearest term.

2.5.2.3 Non-uniform Distributions

Jurie and Triggs show that the distribution of cluster centers is highly non-uniform for dense sampling approaches. When using k-means clustering, high-frequency terms dominate the quantization process, yet these common terms are less discriminative than the medium-frequency terms that can get lost in quantization.Instead, they propose an on-line, fixed-radius clustering method that they demonstrate produces better codebooks of visual elements, meaning that a visual word is more likely to appear in an image if it has appeared once before. Thus visual words are not independent samples of the image.Various weighting functions and strategies are proposed and evaluated.

2.5.2.4 Intracluster Distances

There is a trade-off involved in choosing the granularity of the quantization, where finer-grained clustering leads to potentially more discriminative information being preserved at the cost of increased storage and computational require- ments. A possible compromise is to use a courser-grained quantization, but compensate for the lack of discrimination by employing an efficient method for incorporating intra- cluster distances to weight terms. To this end, Jegou et al. developed a technique called Hamming Embedding, which efficiently represents how far a feature lies from the cluster center and thus how much weight to assign to that term for the detected feature [10].

2.6 Speech Synthesis

Speech synthesis is the artificial production of human speech. A computer system used for this purpose is called a speech synthesizer, and can be implemented in software or hardware. A text-to-speech (TTS) system converts normal language text into speech; other systems render symbolic linguistic representations like phonetic transcriptions into speech [12].

Synthesized speech can be created by concatenating pieces of recorded speech that are stored in a database. Systems differ in the size of the stored speech units; a system that stores phones or diphones provides the largest output range, but may lack clarity. For specific usage domains, the storage of entire words or sentences allows for high-quality output. Alternatively, a synthesizer can incorporate a model of the vocal tract and other human voice characteristics to create a completely ”synthetic” voice output. The quality of a speech synthesizer is judged by its similarity to the human voice and by its ability to be understood. An intelligible text-to-speech program allows people with visual impairments or reading disabilities to listen and written works on a home computer. Many computer operating systems have included speech synthesizers since the early 1990s [13].

A text-to-speech system (or ”engine”) is composed of two parts: a front-end and a back-end. The front-end has two major tasks. First, it converts raw text containing symbols like numbers and abbreviations into the equivalent of written-out words. This process is often called text normalization, pre-processing, or tokenization. The front-end then assigns phonetic transcriptions to each word, and divides and marks the text into prosodic units,

like phrases, clauses, and sentences. The process of assigning phonetic transcriptions to words is called text-to-phoneme or grapheme-to-phoneme conversion. Phonetic transcriptions and prosody information together make up the symbolic linguistic representation that is output by the front-end. The back-end—often referred to as the synthesizer—then converts the symbolic linguistic representation into sound. In certain systems, this part includes the computation of the target prosody (pitch contour, phoneme durations), which is then imposed on the output speech.as shown in Figure. 2.3 [14].

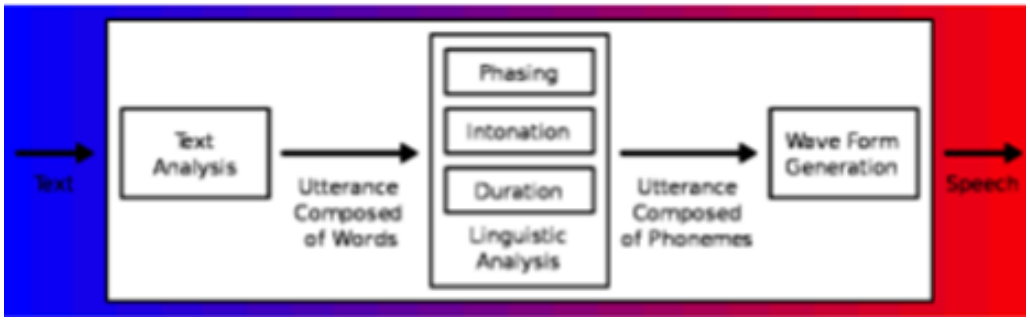

Figure 2.3: Overview of a Typical Text TO Speech System

2.7 Related Studies

Sawant et al., [15], reviewed the research carried out in the sensor based approach and reported that Lee and Xu developed a glove-based gesture recognition system that was able to recognize 14 of the letters from the hand alphabet, learn new gestures and able to update the model of each gesture in the system in online mode, with rate of 10Hz. Over the years advanced glove devices have been designed such as the *Sayre Glove*, *Dexterous Hand Master* and *Power Glove*. The most successful commercially available glove is by far the *VPL Data Glove*. It was developed by Zimmerman during the 1970's. It is based upon patented optical fiber sensors along the back of the fingers and Starner and Pentland developed a glove-environment system capable of recognizing 40 signs from the American Sign Language with a rate of 5Hz [15].

In [16], a sign recognition system has been carried out by using web camera and image processing technology (bounding box function and SIFT a logarithm) with neural network technology for recognition purpose. By using GSM module the system has been turned to an effective communication system and became more friendly users by using GUI. Also, in [8], another

system using Image processing technique (SURF algorithm) has been used to improve the quality of teaching and learning in deaf and dumb institute by recognized 26 alphabets to achieve a good recognition rate along with a low time complexity. The advantages of SIFT and SURF methods (hybrid feature descriptor), has been used as a combined feature set for sign language recognition. To further increase the recognition rate and make the recognition system resilient to view-point variations, the concept of derived features from the available feature set has been introduced in [17]. K-Nearest Neighbor (KNN) and Support Vector Machine (SVM) has been used for hybrid classification of single signed letter. Comparative study of these methods with other popular techniques shows that the real time efficiency and robustness are better.

In [18] nine gestures are recognized and converted to speech and text in real time mode by using MATLAB. YCBCR color transformation is used for feature extraction and PCA algorithm is used for recognition to captured image using web camera. In [19], PCA algorithms also have been used to recognize 26 gesture from Indian sign language, morphological filter and outs algorithms for segmentation to get comparable accuracy. A comparison has been did between using web camera and traditional image processing techniques against android device and PCA algorithms the first method is more accurate (90) But it take time and memory the second method is faster and need less memory but has low accuracy(77) [20].

The study in [1] introduced an intelligent system using image processing (centroid, peak calculation, and angle calculation), data mining and artificial intelligence concepts has been built to take visual inputs of sign languages hand gestures and generate easily recognizable form of outputs in the form of text and voice or vice versa in real time mode with 82 accuracy.

In [2], 24 Different alphabets of Indian Sign Language has been recognized where continuous video sequences of the signs with bare hands have been considered with a success rate of 96.25, by comprised of three stages: Preprocessing stage(skin filtering, histogram matching), Feature Extraction(Eigen values and Eigen Vectors) and Classification (Eigen value weighted Euclidean distance). Another system, in [4], does not require the background to be perfectly black. It works on any background to recognition sign language has been done by using image preprocessing, calculating coordinate for feature extraction and finally the pattern matching algorithm for classifying purpose.

In terms of machine learning-based approaches, Abdo et al, [21], employed Hidden Markov Model (HMM) for classifying letters shape for Arabic alphabet and numbers sign language recognition in real word this system is suitability and reliability compared with other competitive systems but the limitations of this system is required users to wear color glove. Also, in the study by Dogic and karli, [22], sign language recognition has been applied with accuracy 84, the work has been did with the use of digital image processing methods providing a system that teaches a multilayer neural network using a back propagation algorithm. Images has been processed by feature extraction methods (canny edge detector), and by masking method the data set has been created. Training has been did using cross validation method for better performance.

Video processing technique has been used to translate Real time Arabic sign language to Arabic text. For example, the technique used in [23] include video segmentation (shot boundary detection, key frame extraction), pattern construction and discrimination (key frame processing) and feature extraction, the extracted features are intensity histogram and Gray Level Co-Occurrence Matrix (GLCM). To identify English alphabetic sign language without require the hand to be perfectly aligned to the camera, image processing technique (the detection of skin and marker pixels) has been used in [15]. For purpose of segmentation, threshold has been used finally to extract feature and recognition, the coordinate calculation, color calibration and pattern matching algorithm has been used ,this system is easy and high accuracy but it require users to wear special color band in their fingers.

The study in [3] proposes a novel technique that recognizes finger spelled American Sign Language ASL gestures with accuracy 79.9, simple and takes less than a half second for its complete execution So, for real time application this technique is fast enough to provide quick responses. Also, the developed algorithm is rotation, luminance and translation invariance, the 72 point descriptor have been used to extract feature and Euclidean distance to classify. In [24], contour features have been used to recognize the sign language First, the sign language gesture images pre-processing such as background image difference is given. Then the method of extracting edge features and contour characteristics has been introduced. Finally, the match method of weighted features with the sample image feature values has been given to get accuracy 93.

Finally, Otsu's Algorithm is an logarithms for segmentation and canny edge detector for feature extraction they have been used for sign language recognition [5]. The set of 32 signs, each representing the binary 'UP' and 'DOWN' positions of the five fingers has been defined. The images are of the palm side of right hand and have been loaded at runtime i.e. dynamic loading. The method has been developed with respect to single user both in training and testing phase. The static images have been pre-processed using feature point extraction method and have been trained with 10 numbers of images for each sign. The images have been converted into text by identifying the fingertip position of static images using image processing techniques. The proposed method was able to identify the images of the signer which are captured dynamically during testing phase. The Sign Language Recognition System was able to recognize images with 98.125 accuracy when trained with 320 images and tested with 160 images [25].

Table 2.2: Summary Of The Related Study

Author	Year	Title	Algorithms	Drawback	Advantage
S.Pramada, D.Saylee, N.Pranita, N.Samiksh, and M. Vaidya	2013	Intelligent sign language recognition using image processing	calculation coordinate, color calibration and pattern matching algorithm	require users to wear special color band in their fingers	easy and high accuracy
S.ĐogićandG, Karli	2014	Sign language recognition using neural networks	multilayer neural network using a back propagation algorithm and canny edge detector	using web camera and Require black background	accuracy 84
S. A. E.-R. S, Mahmoud Zaki Abdo, Alaa Mahmoud Hamdy, and E. M. Saad	2015	Arabic alphabet and numbers sign language recognition	Hidden Markov Model (HMM)	required users to wear color glove	real time mode, suitability

M. U. Kakde and A. M. Rawate	2016	Hand gesture recognition system for deaf and dumb people using PCA	YCBCR color transformation and PCA algorithm	using web camera	High accuracy
SHWETA SONAJIRAO SHINDE and Dr. R.M. AUTEE	2016	Real Time Hand Gesture Recognition And Voice Conversion System For Deaf And Dumb Person Based On Image Processing	data mining and artificial intelligence concepts	using web camera	real time mode with 82 accuracy

Chapter Three
Algorithmic Design Approach

3.1 Proposed Model

The dataset used in the proposed model was collected[1] and created using phone camera without perfect alignment. First, the system processes the dataset images and apply skin detection algorithm to them and detect the skin color pixels. Then it will make the image binary. Second, for feature extraction and evaluation we are using Bag of features in category classification. It splits the image grid by grid and takes number of image patches from it. The strongest features are identified by using SURF algorithm and K-means clustering is used for vector quantization. The features or bag of words are stored in the feature vector After that for multi-class SVM (Support Vector Machine) classifier has been used for categorize training and testing set for evaluation. Thirdly, the system will take the tested image and apply skin detection algorithm on it for purpose of skin detection, using SURF algorithm to extract features and K-mean clustering for vector quantization finally it was compared with features in dataset to find the corresponding letter. This proposed model is shown in Figure 3.1.

3.2 Dataset Collection

As is not still a widely researched topic we did not find any dataset on any resources. Therefore, we made our own dataset. We took images of hands for males and females in different ages and with different positions also in different background using camera phone and without perfectly aligned with camera. We acquired images for 26 alphabet of American Sign Language. We took images of 10 people's hand in different positions. So in total in the dataset we have 260 images. For each alphabet we are getting ten images of ten people in different postures and different ages.

[1]Datasets were collected manually by the thesis' author.

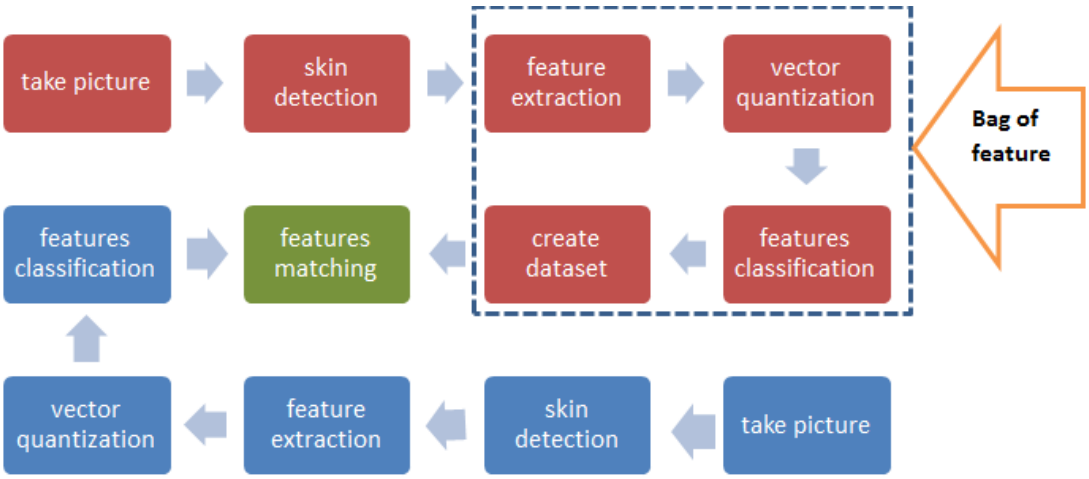

Figure 3.1: Proposed Model

3.3 Skin Color Detection Algorithm

Skin color detection was done to the input image for detection of hand gestures. This technique was used for separating the skin colored regions from the non-skin colored regions. The steps used in this skin color detection algorithm are shown in Figure 3.2.

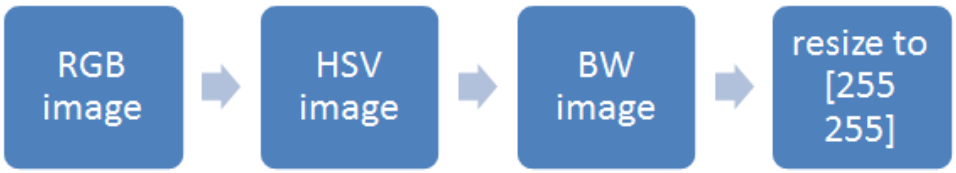

Figure 3.2: Skin Color Detection Algorithm

RGB image is converted to HSV color space which was used because it is more convenient for research purpose Conceptually, the HSV color space is a cone. Viewed from the circular side of the cone, the hues are represented by the angle of each color in the cone relative to the 0° line, which is traditionally assigned to be red. The saturation is represented as the distance from the center of the circle. Highly saturated colors are on the outer edge of the cone, whereas gray tones (which have no saturation) are at the very center. The brightness is determined by the colors vertical position in the cone. At the pointy end of the cone, there is no brightness, so all colors are black. At the fat end of the cone are the brightest colors

As hue varies from 0 to 1.0, the corresponding colors vary from red through yellow, green, cyan, blue, magenta, and back to red, so that there are actually red values both at 0 and 1.0. As saturation varies from 0 to 1.0, the corresponding colors (hues) vary from unsaturated (shades of gray) to fully saturated (no white component). As value, or brightness, varies from 0 to 1.0, the corresponding colors become increasingly brighter. As shown in Figure 3.3 [9].

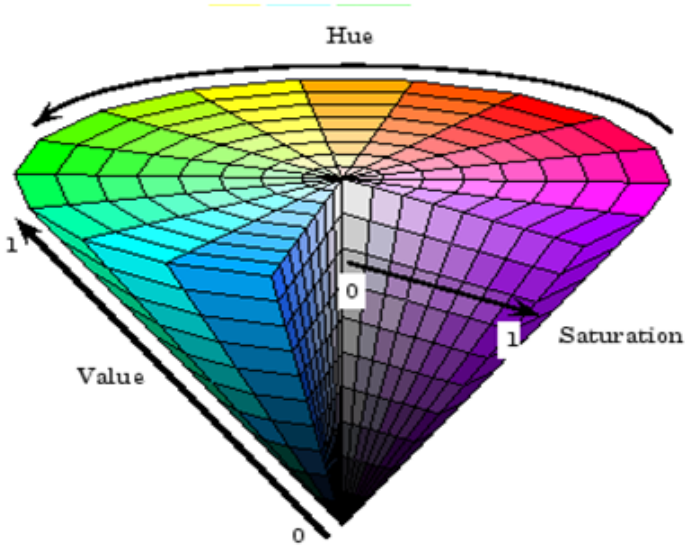

Figure 3.3: HSV Color Space

The conversion of RGB to HSV is given by

$$H = \arccos \frac{\frac{1}{2}(R-G)+(R-B)}{\sqrt{(R-G)^2+(R-B)(G-B)}}, \tag{3.1}$$

$$S = (1-3)\frac{\min(R,G,B)}{R+G+B}, \tag{3.2}$$

$$V = \frac{1}{3(R+B+G)}. \tag{3.3}$$

People have different color skin so the value of H, S and V are different from person to person depend on his color skin, the images in dataset was taken from different persons has different color skin, the value of H, S and V are set manually for all images after that the image converted to black and white form and resized to [255 255] pixels.

So in this research SURF algorithm was used for feature detection and feature descriptor.

3.4 Bag of Feature Approach

3.4.1 Feature Detector and Descriptor: SURF Algorithm

SURF is a local feature detector and descriptor that can be used for tasks such as object recognition or registration or classification or 3D reconstruction. It is a scale and in-plane rotation invariant feature. The detector locates the interest points in the image, and the descriptor describes the features of the interest points and constructs the feature vectors of the interest points. SURF features are invariant of shifting, rotation and scaling, and partially invariant to illumination and affine transformation [26].

SURF approximates the DoG with box filters. Instead of Gaussian averaging the image, squares are used for approximation since the convolution with square is much faster if the integral image is used. Also this can be done in parallel for different scales. The SURF uses a BLOB detector which is based on the Hessian matrix to find the points of interest. For orientation assignment, it uses wavelet responses in both horizontal and vertical directions by applying adequate Gaussian weights. For feature description also SURF uses the wavelet responses. A neighborhood around the key point is selected and divided into subregions and then for each subregion the wavelet responses are taken and represented to get SURF feature descriptor. The sign of Laplacian which is already computed in the detection is used for underlying interest points. The sign of the Laplacian distinguishes bright blobs on dark backgrounds from the reverse case. In case of matching the features are compared only if they have same type of contrast (based on sign) which allows faster matching [11]. Algorithm consists of four main parts:

1. Integral image generation,
2. Fast-Hessian detector (interest point detection),
3. Descriptor orientation assignment (optional),
4. Descriptor generation [26].

3.4.1.1 SURF Detector

The SURF detector focuses its attention on blob like structures in the image. These structures can be found at corners of objects, but also at the location where the reflection of light in the specular surface are maximal The SURF detector algorithm can thus be summarized by the following steps: as shown

in Figure 3.4.

1. Form the scale-space response by convolving the source image using DoH filters,
2. Search for local maxima across neighbouring pixels and adjacent scales within different octaves,
3. Interpolate the location of each local maxima found,
4. For each point of interest, return the DoH magnitude, and the Laplacian's sign [26].

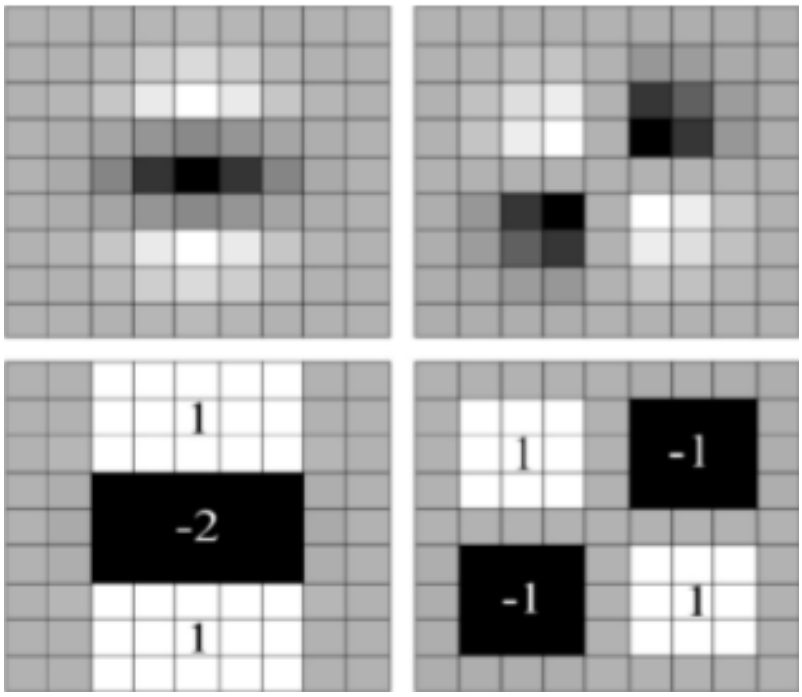

Figure 3.4: Discretized and cropped Gaussian second order partial derivatives in y direction and xy-direction, and their approximations using box filter

3.4.1.2 SURF Descriptor

To describe each feature, SURF summarizes the pixel information within a local neighbourhood. The first step is determining an orientation for each feature, by convolving pixels in its neighbourhood with the horizontal and the vertical Haar wavelet filters. Shown in Figure 3.5, these filters can be thought of as blockbased methods to compute directional derivatives of the image's intensity. By using intensity changes to characterize orientation, this

descriptor is able to describe features in the same manner regardless of the specific orientation of objects or of the camera. This rotational invariance property allow SURF features to accurately identify objects within image taken from different perspective [26].

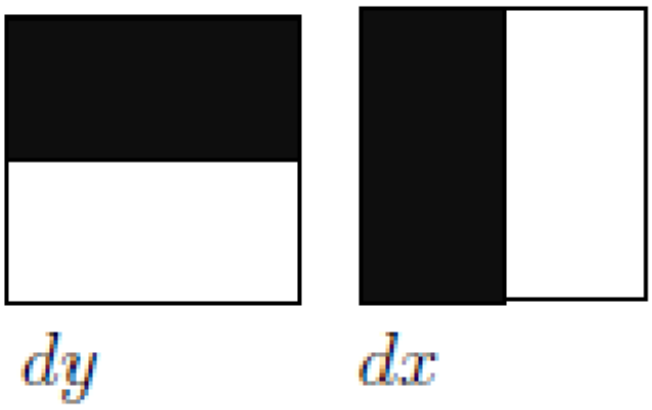

Figure 3.5: The wavelets response. Black and white areas corresponds to a weight-1 and 1 for the Haar kernels

3.4.2 Quantization and Distance Measures

Vector Quantization (Clustering) is used to build the visual vocabulary in Bag of Features algorithms. Nearest-neighbor assignments are used not only in the clustering of features but also in the comparison of term vectors for similarity ranking or classification.Many BoF implementations are described as using K-means cluster, so it was used in this research [10].

3.4.2.1 k-Means Clustering

A key development in image classification using key points and descriptors is to represent these descriptors using a BoW model. Although spatial and geometric relationship information between descriptors is lost using this assumption, the inherent simplification gains make it highly advantageous. The descriptors extracted from the training images are grouped into N clusters of visual words using K-means. A descriptor is categorized into its cluster centroid using a Euclidean distance metric. For our purposes, we choose a value of $N = 500$. This parameter provides our model with a balance. For a query image, each extracted descriptor is mapped into its nearest cluster centroid. A histogram of counts is constructed by incrementing a cluster centroid's

number of occupants each time a descriptor is placed into it. The result is that each image is represented by a histogram vector of length N. It is necessary to normalize each histogram by its L2-norm to make this procedure invariant to the number of descriptors used. Applying Laplacian smoothing to the histogram appears to improve classification results. With K-means clustering process we want to minimize sum of squared Euclidean distances between points x_i and their nearest cluster centers m_k [10].

$$D(x, m) = sum(cluster) - k * sum(\text{point i in cluster}) - k\ (x_i - m_k)^{\ 2} \qquad (3.4)$$

The steps of the algorithm are as follows.

1. Select initial centroids at random,
2. Assign each object to the cluster with the nearest centroid,
3. Compute each centroid as the mean of the objects assigned to it,
4. Repeat previous 2 steps until no change.

In Figure 3.6 (a), (b) and (c) the step by step process of clustering is shown.

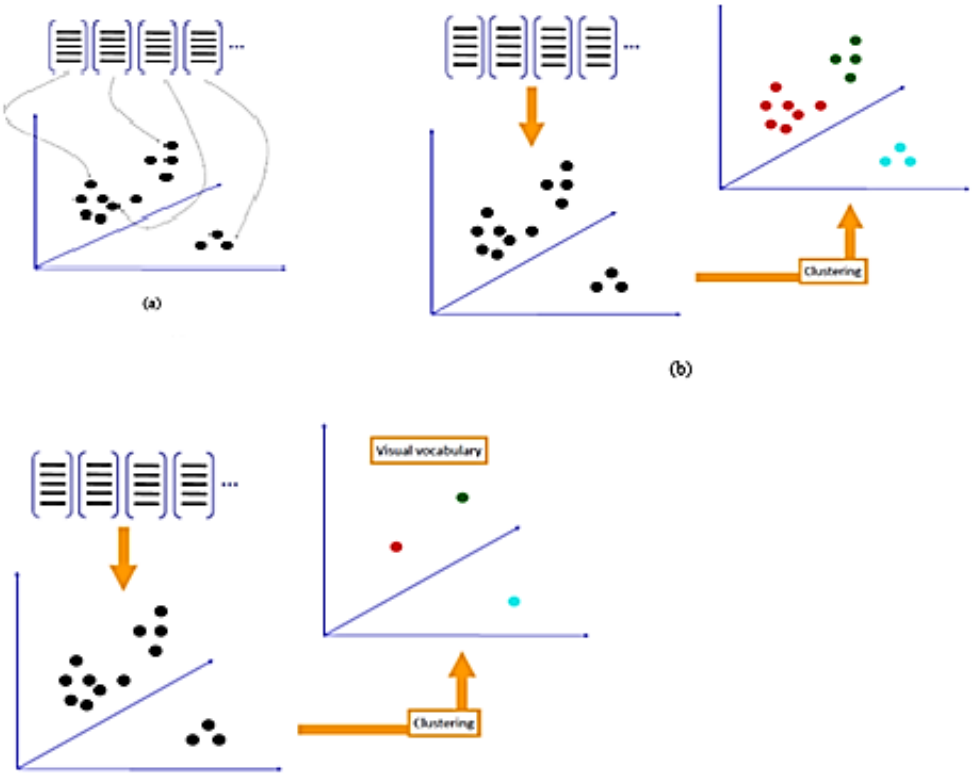

Figure 3.6: The Step By Step Process Of Clustering

3.4.3 Feature Classification

For training the dataset we used SVM (Support Vector Machine) classifier. Support vector Machines are a set of supervised learning methods used for classification, regression and outlier's detection. This is a very popular classifier in gesture recognition. It is very effective in high dimensional spaces. If the number of dimensions is greater than sample number in those cases it is still very operative. It is memory efficient. Different kernel functions can be identified for the decision function [26].

3.4.4 Support Vector Machine

Support Vector Machine (SVM) is primarily a classifier that performs classification tasks by constructing hyper planes in a multidimensional space separating cases of different class labels. According to SVM the decision boundary should be as far away from the data of both classes as possible. Let us consider that we have data points belonging to two classes, A-and A+. Each point in the dataset comes with a class label y, +1 or 1, indicating one of two classes A- and A+ as shown in figure.3.7.

Let us start with a strictly linearly separable case, i.e. there exists a hyper plane which can separate the data A-and A+. In this case we can separate the two classes by a pair of parallel bounding planes.

$$\mathbf{w}^T\mathbf{x} + b = +1, \tag{3.5}$$

$$\mathbf{w}^T\mathbf{x} + b = -1, \tag{3.6}$$

where $\mathbf{w}$ is the normal vector to these planes and b determines their location relative to the origin. The first plane of equation bounds the class A+ and the second plane bounds the class A-. SVM achieves better prediction ability via maximizing the margin between two bounding planes. Hence, SVM searches for a separating hyper plane by maximizing$\frac{2}{\|w\|^2}$ .The linear separating hyper plane is the plane:

$$\mathbf{w}^T\mathbf{x} + b = 0 \tag{3.7}$$

By maximizing the margin between the bounding planes we get an optimal solution:

$$\Big(\mathbf{w}^*, b^*\Big). \tag{3.8}$$

The data points on the bounding planes $\mathbf{w}^{*T}\mathbf{x} + b^* = \pm 1$ are called support vectors. Once we have the training result, all we need to keep in our databases

are the support vectors. If the classes are linearly inseparable then the two planes bound the two classes with a soft margin.

Many datasets cannot be well separated, even after using a soft margin, by a linear separating hyper plane, but could be linearly separated if mapped into a higher or much higher dimensional space by using a nonlinear map. Rather than mapping individual features to higher dimension. Kernel method is used for this purpose. The kernel computes the dot product which would otherwise be much more expensive to compute explicitly. Some widely used Kernel functions are Linear, Polynomial, Radial Basis Function (RBF) and Multi-Layer Perceptron (MLP). We will be using RBF [26].

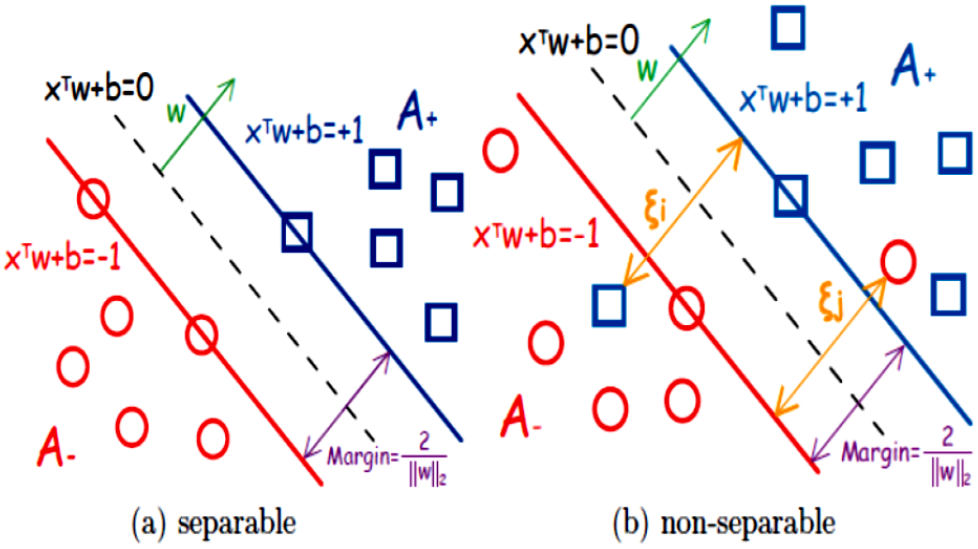

Figure 3.7: Linear SVM For Separable And Non Separable Data

3.5 Graphical User Interface

A graphical user interface (GUI) is a pictorial interface to a program. A good GUI can make programs easier to use by providing them with a consistent appearance and with intuitive controls like pushbuttons, list boxes, sliders, menus, and so forth. The GUI should behave in an understandable and predictable manner.

A graphical user interface provides the user with a familiar environment in which to work. This environment contains pushbuttons, toggle buttons, lists, menus, text boxes, and so forth. The three principal elements required to create a MATLAB Graphical User Interface are:

1. Components: Each item on a MATLAB GUI (pushbuttons, labels, edit boxes, etc.) is a graphical component. The types of components include graphical controls (pushbuttons, edit boxes, lists, sliders, etc.), static elements (frames and text strings), menus, and axes. Graphical controls and static elements are created by the function uicontrol, and menus are created by the functions uimenu and uicontextmenu. Axes, which are used to display graphical data, are created by the function axes.

2. Figures: The components of a GUI must be arranged within a figure, which is a window on the computer screen. In the past, figures have been created automatically whenever we have plotted data.

3. Callbacks: Finally, there must be some way to perform an action if a user clicks a mouse on a button or types information on a keyboard. A mouse click or a key press is an event, and the MATLAB program must respond to each event if the program is to perform its function. For example, if a user clicks on a button, that event must cause the MATLAB code that implements the function of the button to be executed. The code executed in response to an event is known as a call back. There must be a callback to implement the function of each graphical component on the GUI.

MATLAB GUIs are created using a tool called guide, the GUI Development Environment. This tool allows a programmer to layout the GUI, selecting and aligning the GUI components to be placed in it. Once the components are in place, the programmer can edit their properties: name, color, size, font, text to display, and so forth. When guide saves the GUI, it creates working program including skeleton functions that the programmer can modify to implement the behavior of the GUI.When guide is executed, it creates the Layout Editor, shown in Figure.3.8. The large white area with grid lines is the layout area, where a programmer can layout the GUI. The Layout Editor window has a palate of GUI components along the left side of the layout area. A user can create any number of GUI components by first clicking on the desired component, and then dragging its outline in the layout area. The top of the window has a toolbar with a series of useful tools that allow the user to distribute and align GUI components, modify the properties of GUI components, add menus to GUIs, and so on [27].

The basic steps required to create a MATLAB GUI are:

1. Decide what elements are required for the GUI and what the function of each element will be.

2. Use a MATLAB tool called guide (GUI Development Environment) to layout the Components on a figure. The size of the figure and the alignment and spacing of components on the figure can be adjusted using the tools built into guide.

3. Use a MATLAB tool called the Property Inspector (built into guide) to give each component a name (a "tag") and to set the characteristics of each component, such as its color, the text it displays, and so on.

4. Save the figure to a file. When the figure is saved, two files will be created on disk with the same name but different extents. The fig file contains the actual GUI that you have created, and the M-file contains the code to load the figure and skeleton call backs for each GUI element.

5. Write code to implement the behavior associated with each callback function [27].

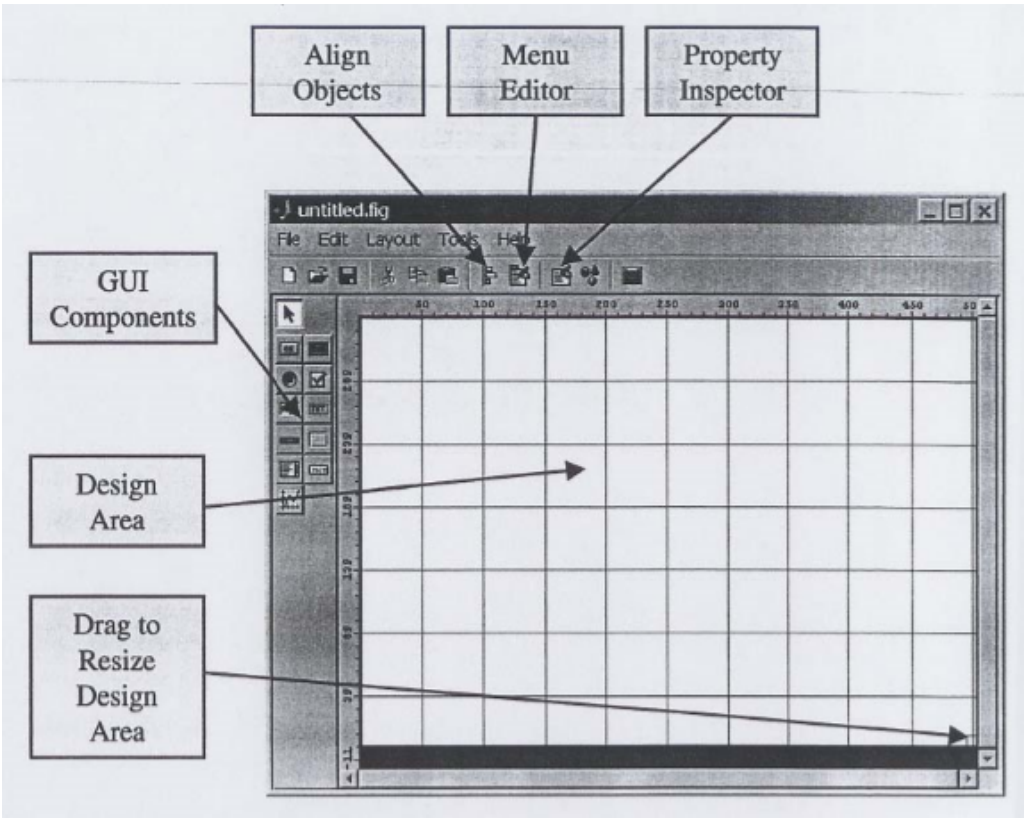

Figure 3.8: The guide tool window

3.6 Implementation Of Hand Gesture Recognition System

Firstly the 260 pictures, 10 images for 26 American sign letter, are taken from different persons with different skin color using phone camera in different backgrounds and different postures without perfect aligned with camera to create dataset, as shown in Figure 3.9.

Figure 3.9: Examples of Pictures for ASL

Secondly to detect skin color, all images were converted to HSV color space, the values of H, S and V were set depending on the color skin of individual,according to these values the hand was segmented from background after that the images was converted to black and white format by using color threshold APP, this process is shown in Figure 3.10 and Figure 3.11. The skin detected image is resized to 255×255 pixels for faster computing.

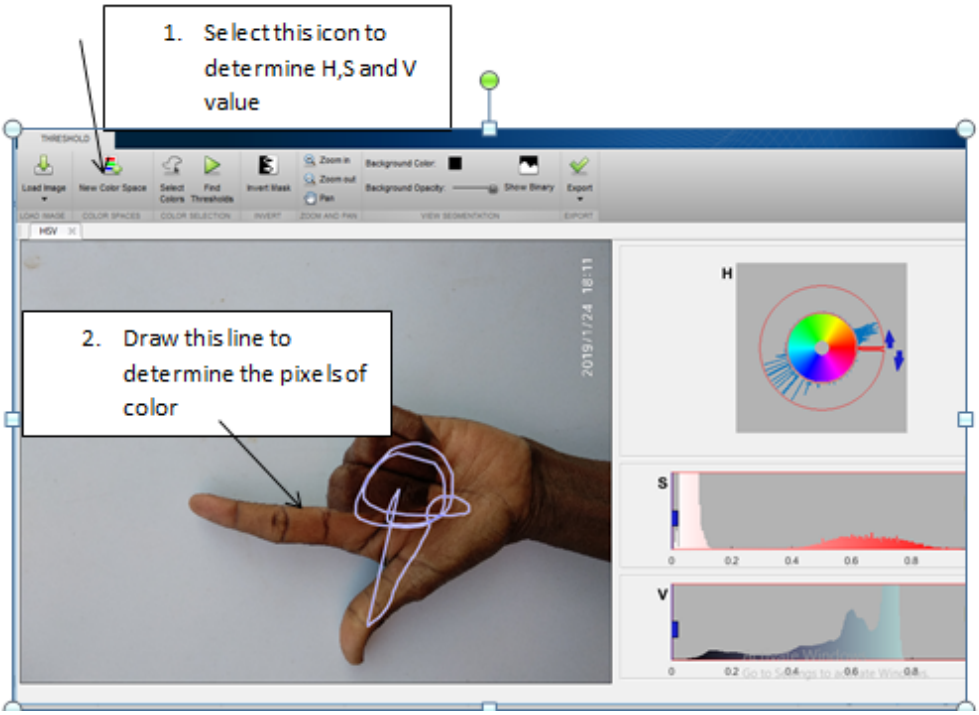

Figure 3.10: The Steps of Determine the H,S and V value for Individual

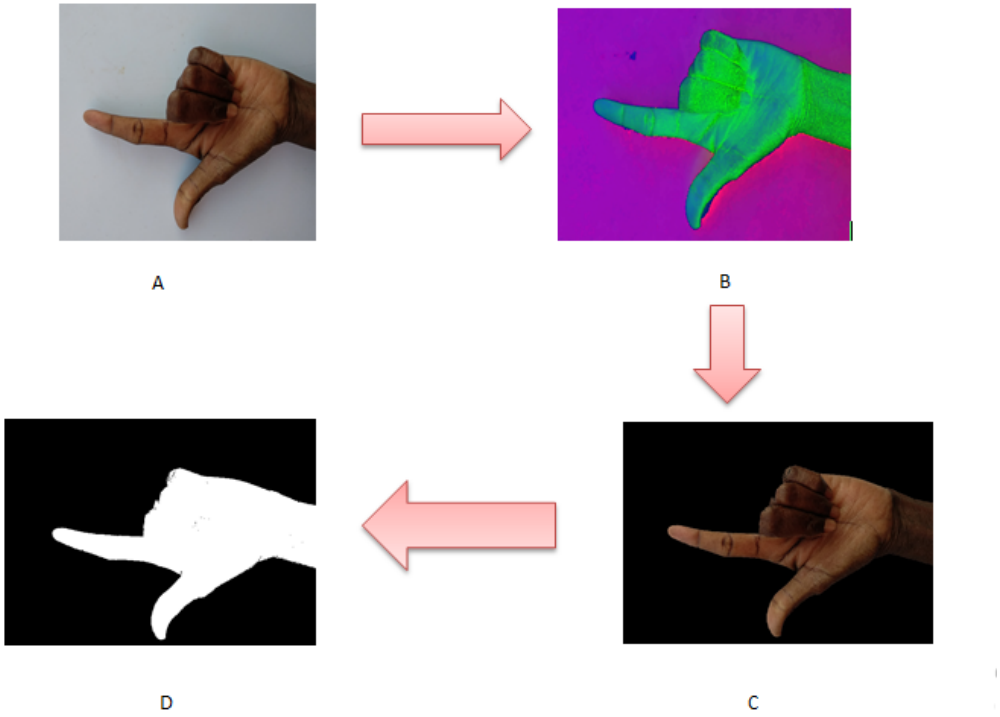

Figure 3.11: Step by step skin detection and binarization: (A) Actual image, (B) Converted to HSV, (c) segmented image, (d) converted to BW format

Finally the bag of feature is used to training dataset and classify image to 26 class according to American sign language alphabets

3.6.1 Steps of Bag of Feature

3.6.1.1 Step 1 : Setting Up Image Category Sets

The images is Organized and partitioned into training and test subsets. the image Set function is used to organize categories of images to use for training an image classifier. Images are Organized into categories made handling large sets of images much easier. the sets are Separated into training and test image subsets. In this research, (90 percent) of the images are partitioned for training and the remainder for testing because this present was given the most accuracy evaluating value as shown in Table 3.1.

Table 3.1: Comparison Between Different Percent

Percent	Accuracy
90	0.89
70	0.87
50	0.65
30	0.56

3.6.1.2 Step 2: Creating Bag of Features

A visual vocabulary, or bag of features is Created, by extracting feature descriptors from representative images of each category.

The bag of Features object defines the features, or visual words, by using the k-means clustering algorithm on the feature descriptors extracted from training Sets. The algorithm iteratively groups the descriptors into k mutually exclusive clusters. The resulting clusters are compacted and separated by similar characteristics. Each cluster center is represented a feature, or visual word.

Speeded up robust features (or SURF) detector is used that is provides greater scale invariance , SURF algorithm is used as detector and descriptor. This process is shown in Figure 3.12; the Surf feature for one image in dataset is shown in Figure 3.13

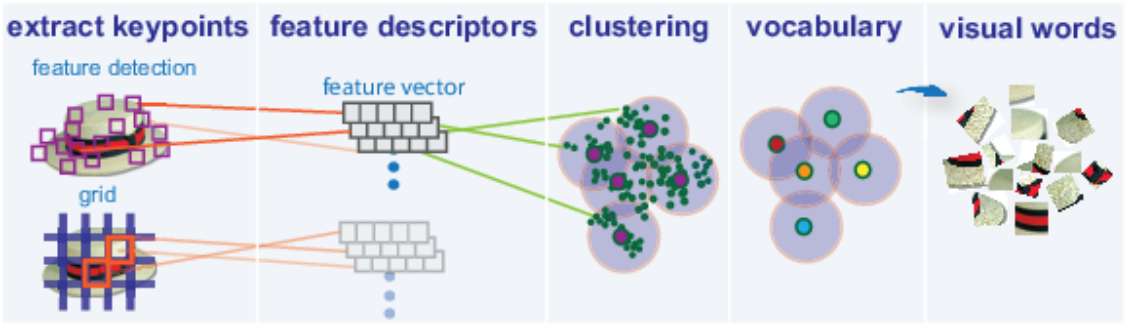

Figure 3.12: Step2: Creating Bag of Feature

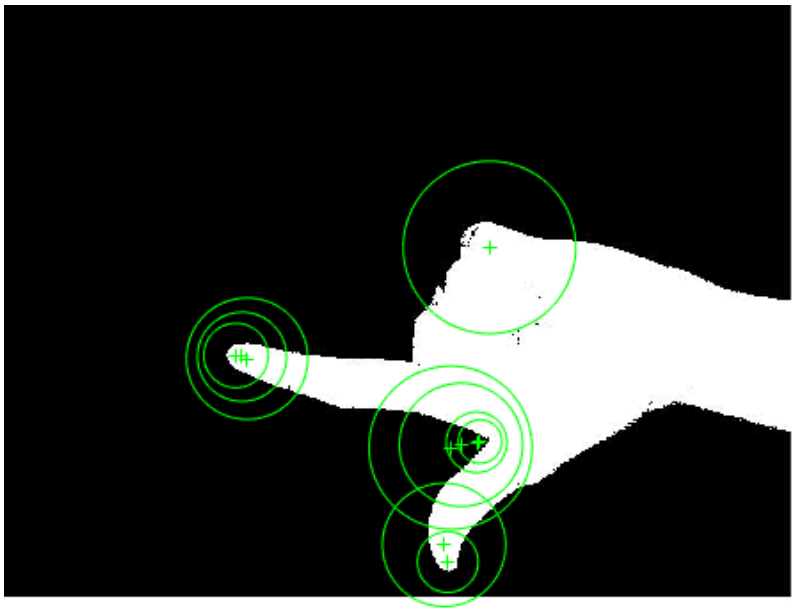

Figure 3.13: SURF Feature for one image from dataset

3.6.1.3 Step 3: Training an Image Classifier With Bag of Visual Words

The train image category classifier function is returned an image classifier. The function is trained a multi class classifier using the error-correcting output codes (ECOC) framework with binary support vector machine (SVM) classifiers.

The train Image Category Classifier function is used the bag of visual words returned by the bag Of Features object to encode images in the image set into the histogram of visual words. The histogram of visual words are then used as the positive and negative samples to train the classifier.

The bag of Features is used to encode each image from the training set. This function detected and extracted features from the image and then used

the approximate nearest neighbor algorithm to construct a feature histogram for each image. The function then incremented histogram bins based on the proximity of the descriptor to a particular cluster center. The histogram length corresponded to the number of visual words that the bag Of Features object constructed. The histogram became a feature vector for the image. This process is repeated for each image in the training set to create the training data. The image Category Classifier is used to evaluate method to test the classifier against the validation image set. The output confusion matrix represented the analysis of the prediction. This process is shown in Figure 3.14 and the feature histogram for one image from data set is shown in Figure 3.15.

Figure 3.14: Step 3: Train an Image Classifier With Bag of Visual Words

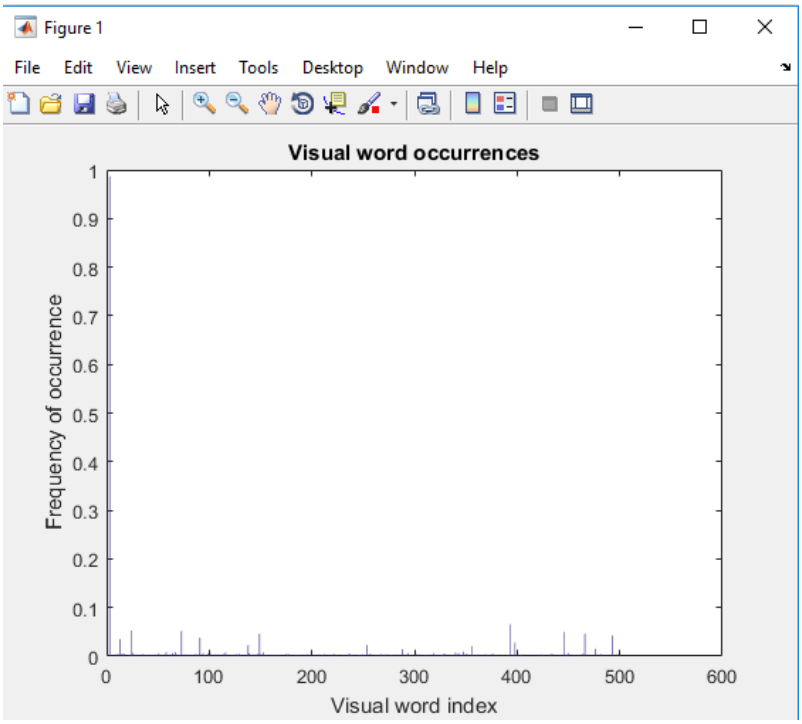

Figure 3.15: Feature Histogram for One Image from data set

3.6.1.4 Step 4: Classifying an Image or Image Set

Image for test is taken by phone camera, color threshold APP is used to convert image to HSV color space and the value of H, V and S are set according to color skin of person , the hand is segmented according to these value , segmented image converted to black and white format , resize to [255, 255] pixels , finally the image Category Classifier predict method is used on the tested image to determine its category, as shown in figure.3.16 below.

After the bag of feature was used to determine the class of image, the letter which corresponding to the image is appeared in workspace, finally the letter is converted to the voice by using SpeechSynthesizer which is included in computer operating systems and implemented in MATLAB by using the code in Listing 1 and as shown in Figure.3.17.

Listing 1: Creating Dynamic Dimensions

```
NET.addAssembly('system.speech');
mySpeaker=System.Speech.Synthesis.SpeechSynthesizer;
determined the rate  and the volume by these command:
mySpeaker.Rate=0;
mySpeaker.Volume=100;
```

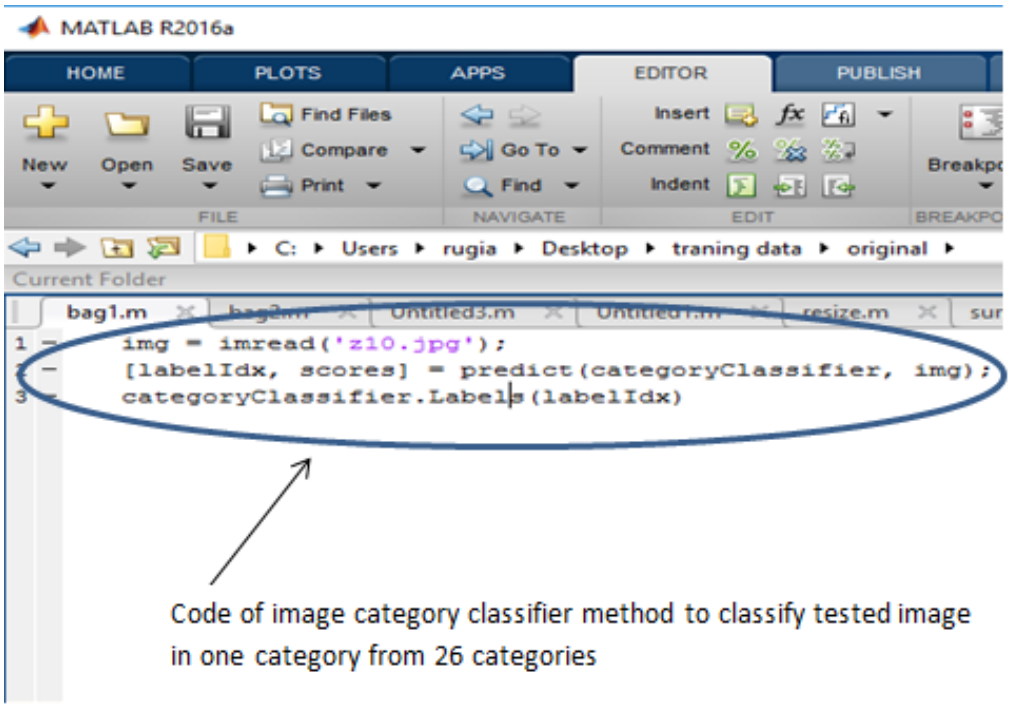

Figure 3.16: Step4: classify tested image

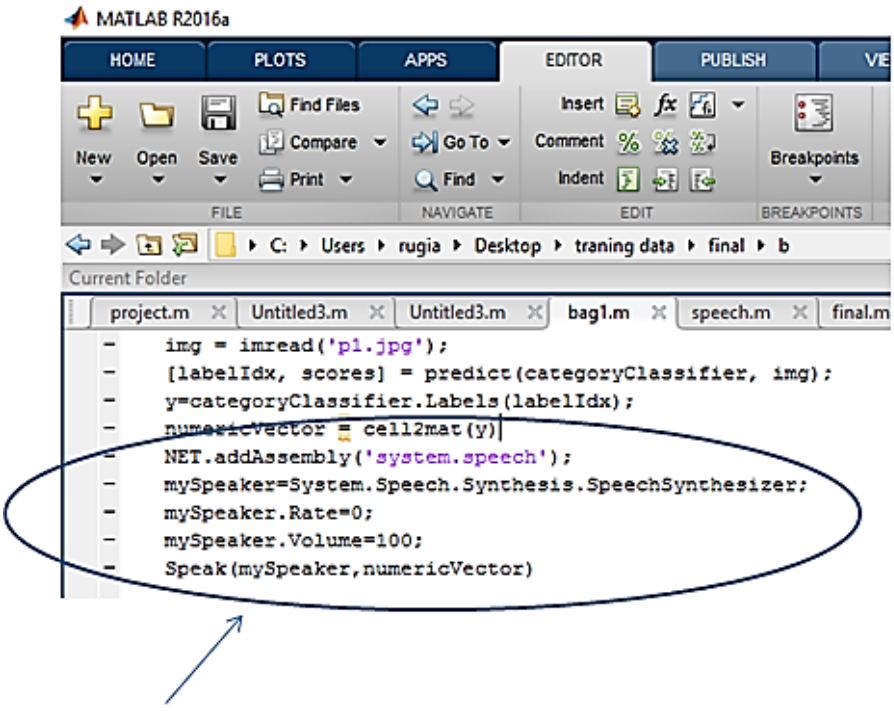

Figure 3.17: The Code of SpeechSynthesizer

Finally to make the system friendlier with user it is converted to graphical user interface was implemented using MATLAB2016 Figure 3.18 is shown the guide tool window for the system.

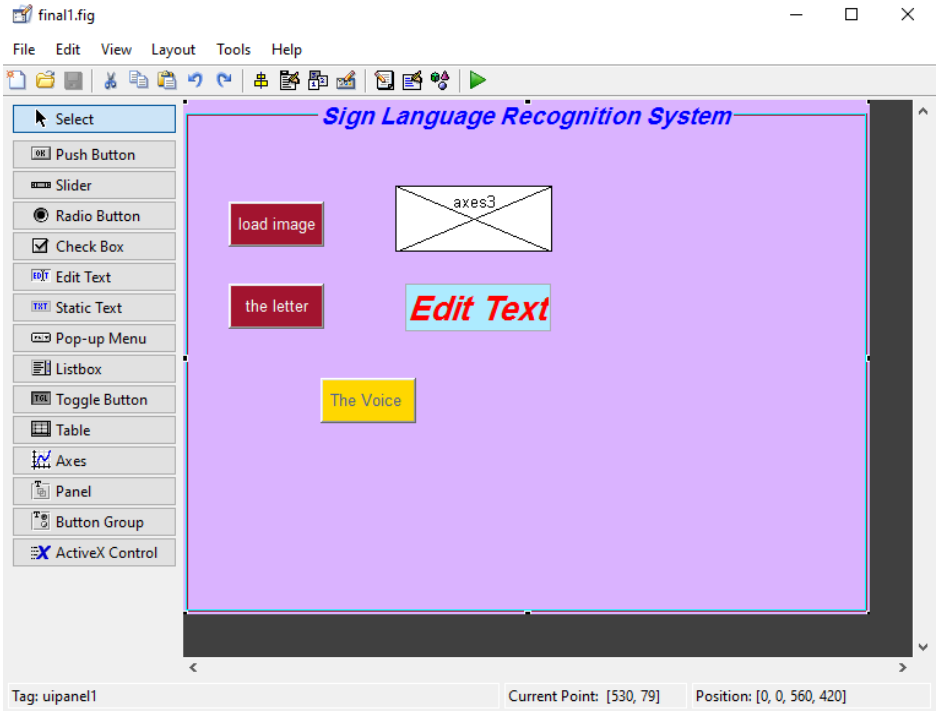

Figure 3.18: The Guide Tool Window for The System

We have taken one axes control to show captured image. We have created the push button to load the captured image through phone camera and also added the push button to classify the image and determined the corresponding letter of the gesture and showed in static text box, another push button is added to convert text to speech Figure 3.19 shown the initial output window.

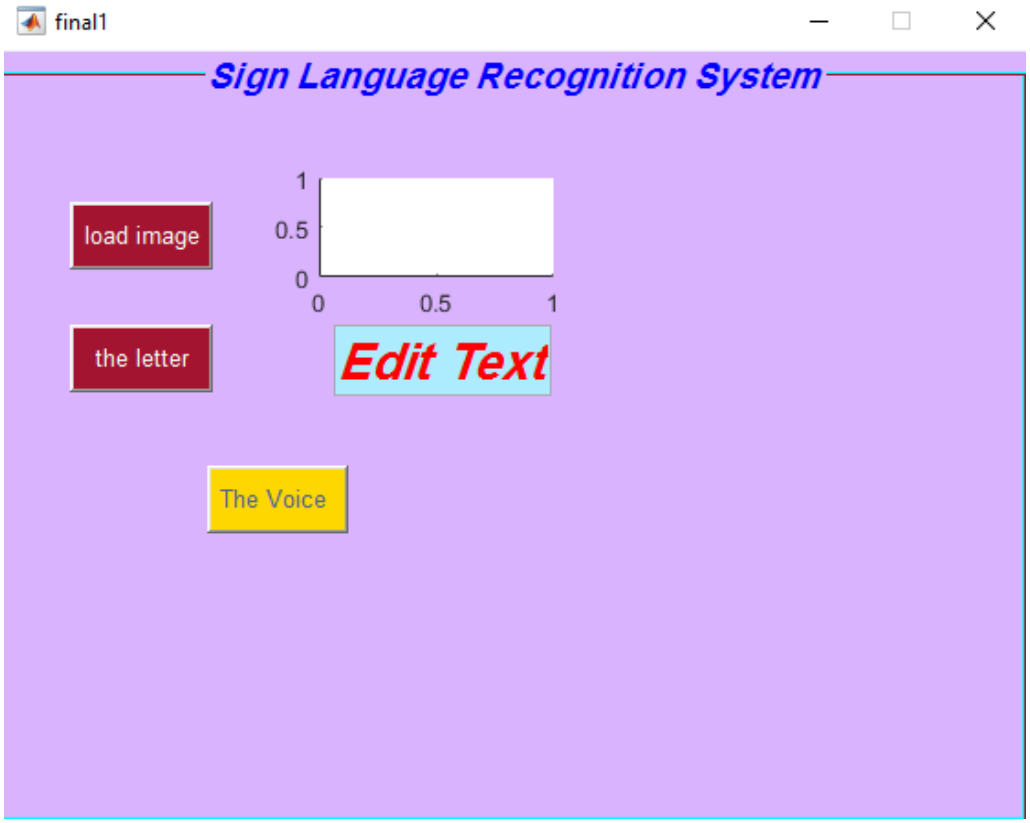

Figure 3.19: The Initial Output Window

Chapter Four
Results and Discussion

4.1 Test and Train Sets

In our proposed system for training we have taken ninety percentages of images for training and the other ten percent for testing from each class randomly. This has been done in the sake of more accurate evaluation as mention in chapter 3.

We found the more images in training set, the more accuracy we have. However, for technical constraints we cannot increase the number of images more than ten in each class. Consequently, the total number of images in training set is 260. as shown in table.4.1 below

Table 4.1: Training Sets

Total Class of images	26
In one class number of images	10
Total images	260
For training the image classifier number of images	234
For testing and evaluation number of images	26

4.2 Evaluating The Model

Train set average accuracy after training the SVM classifier with the train set we evaluated the trained classifier on the train set and got 89 percent average accuracy. The confusion matrix for this test set is shown in Figure 4.1

4.3 Testing The Model

After training the classifier we tested the program on ten people, so there are 260 images, ten images in each class. The photo is taken from individual, loaded in MATLAB, resizing to [255 255] pixels, segmented using color thresh-

old depending on his color, converted to black and white , extracted features and described using SURF algorithm and quantized using K-mean cluster then These preprocessed result compared with stored database. If features of processed image are matched with stored image then text is displayed.

	PREDICTED																								
Known	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
B	0.89	0.00	0.00	0.00	0.00	0.11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
C	0.00	0.00	0.89	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
D	0.00	0.00	0.00	0.89	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
E	0.11	0.00	0.00	0.00	0.78	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.11	0.00	0.00	0.00	0.00	0.00	0.00	0.00
F	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
G	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
H	0.00	0.00	0.00	0.00	0.00	0.33	0.00	0.67	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
I	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
J	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.78	0.00	0.00	0.00	0.00	0.11	0.00	0.00	0.00	0.00	0.11	0.00	0.00	0.00	0.00	0.00	0.00
K	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.89	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.11	0.00	0.00	0.00	0.00	0.00	0.00	0.00
L	0.00	0.00	0.00	0.00	0.00	0.11	0.00	0.00	0.00	0.00	0.00	0.89	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
M	0.00	0.00	0.00	0.00	0.00	0.00	0.11	0.00	0.00	0.00	0.00	0.00	0.67	0.22	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
N	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.78	0.11	0.00	0.00	0.00	0.00	0.11	0.00	0.00	0.00	0.00	0.00	0.00
O	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
P	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Q	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
R	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
S	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
T	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00
U	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00
V	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
W	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00
X	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00
Y	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
Z	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Figure 4.1: The Confusion Matrix for the Test Set

4.3.1 Determining The Number of Images in Training Set

There are 26 classes of signs. Firstly in each class we used five images, the total number of images in training set was 130. For testing module we used

260 images, ten images for each letter.

Table 4.2: Testing Case 1

Number of correct letters	150
Number of wrong letters	110
Total number of tested images	260
For training the image classifier number of images	130

$$\text{Average accuracy} = \frac{\text{correct images}}{\text{total images}} \times 100 \tag{4.1}$$

$$= \frac{150}{260} \times 100 = 57.7\% \tag{4.2}$$

In the second time we increased the numbers of images to seven for each class, the total number of images was 182. For testing module we used 260 images, ten images for each letter

Table 4.3: Testing Case 2

Number of correct letters	106
Number of wrong letters	154
Total number of tested images	260
For training the image classifier number of images	182

$$\text{Average accuracy} = \frac{154}{260} \times 100 = 59.2\% \tag{4.3}$$

Finally we used ten images for each class, and the total number was 260. For testing module we used 260 images, ten images for each letter.

Table 4.4: Testing Case 3

Number of correct letters	220
Number of wrong letters	40
Total number of tested images	260
For training the image classifier number of images	260

$$\text{Average accuracy} = \frac{220}{260} \times 100 = 84.6\% \tag{4.4}$$

From above calculation, the ten images in each class got the greatest average accuracy for our system. Table 4.5 below illustrate the accuracy for each try in all classes by detail.

Table 4.5: The Accuracy of Module

The letter	10 images	7 images	5 images
A	100	50	80
B	70	70	50
C	90	90	80
D	70	40	30
E	70	70	30
F	90	30	50
G	100	50	80
H	90	40	50
I	90	40	30
J	70	70	70
K	100	70	50
L	100	60	80
M	70	60	40
N	50	50	40
O	100	60	100
P	90	70	60
Q	100	70	80
R	90	50	60
S	70	50	60
T	50	60	40
U	100	70	50
V	80	40	40
W	100	90	80
X	70	60	40
Y	90	60	70
Z	90	70	60

Figure 4.2 the accuracy for each class was plotted in line for all tries and in Figure 4.3 the accuracy for each class was plotted again in column for

more clear vision, it is obvious when ten images are used we got the greatest accuracy in all class when used the Seven images, the accuracy is decreased in all class when Five images are used the accuracy is decreased in 17 class and increased in 9 class it is obvious from above discussion, figures and table, the more images in training set, the more accuracy we have.

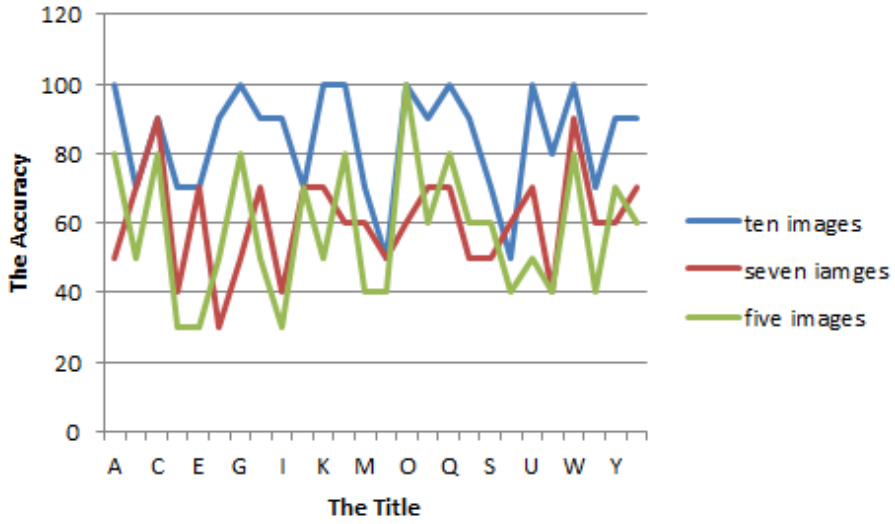

Figure 4.2: The Accuracy of Module in Line:Blue:Ten Images, Red:Seven Images, Green:Five Images

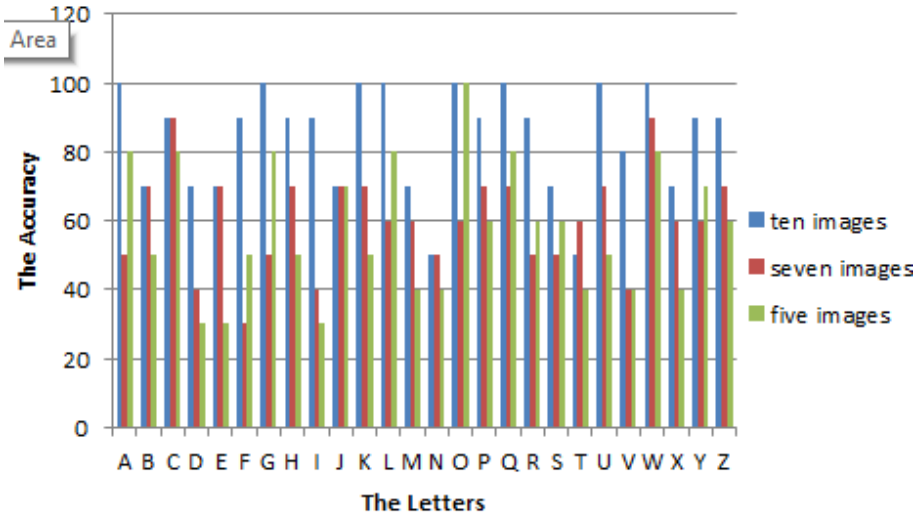

Figure 4.3: he Accuracy of Module in Column:Blue:Ten Images, Red:Seven Images, Green:Five Images

4.3.2 Analysis and Discussion of Result

The average accuracy for the model we received is eighty-five percentages when use ten images for each class in training set for these reason we were used it in our model instead of five or seven images, the total images in training set are 260. By Using 260 images for testing the module , we got the results that is illustrate in details for each letters in Figure 4.4 and the accuracy with shape of sign letter for each letters in Figure 4.5.

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
A	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
B	✓	✓	✓	✓	✓	×	×	✓	×	✓
C	✓	✓	✓	✓	✓	✓	✓	×	✓	✓
D	✓	✓	×	✓	×	✓	✓	✓	✓	✓
E	✓	×	×	✓	✓	✓	✓	×	✓	✓
F	✓	✓	✓	✓	✓	✓	✓	✓	×	✓
G	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
H	✓	×	✓	✓	✓	✓	✓	✓	✓	✓
I	✓	✓	✓	×	✓	✓	✓	✓	✓	✓
J	✓	×	×	✓	✓	✓	✓	✓	×	✓
K	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
L	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
M	×	✓	×	×	✓	✓	✓	✓	✓	✓
N	×	✓	×	✓	✓	×	✓	✓	×	×
O	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
P	✓	✓	✓	×	✓	✓	✓	✓	✓	✓
Q	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
R	✓	✓	✓	✓	×	✓	✓	✓	✓	✓
S	✓	×	✓	×	×	✓	✓	✓	✓	✓
T	×	✓	×	✓	✓	✓	✓	×	×	×
U	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
V	✓	✓	✓	✓	✓	✓	×	✓	✓	×
W	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
X	✓	✓	✓	×	✓	✓	✓	×	✓	×
Y	×	✓	✓	✓	✓	✓	✓	✓	✓	✓
Z	✓	×	✓	✓	✓	✓	✓	✓	✓	✓

Figure 4.4: Results of Each Letters in Detail

Figure 4.5: The Accuracy of letters : Red:100, Black:90, Blue:80, Yellow:70, Green:50

In Figure 4.6 the accuracy for each class was plotted in column and in Figure 4.7 the accuracy for each class was plotted again in line. The accuracy rate fluctuates between 50 percent and 100 percent. There are eight letters have 100 percent of accuracy (A, G, K, l, O, Q, U and W).It is noticeable from figure.4.5 that these letters have distinguished shape from the other letters. That is why they are the most accurate ones. The letters C, F, H, I, P, R, Y and Z have 90 percent of accuracy. It was also a great accuracy because these letters have distinguished shape from the other letters too, but due to the limited resolution of phone camera beside the orientation of image is not standard, the accuracy of these letters degraded to 90 percent.

Figure 4.6: The Accuracy of the Module in Line

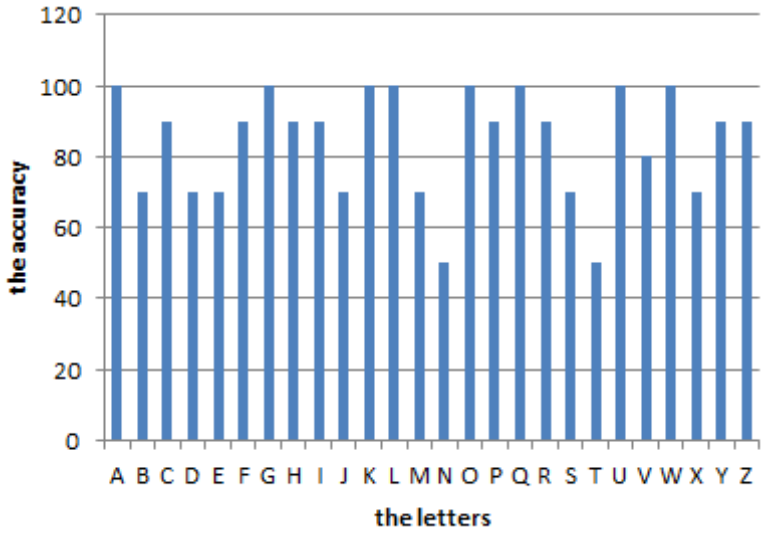

Figure 4.7: The Accuracy of the Module in Column

The only one letters with 80 percent of accuracy was V. it has shape approximately close to the shape of K letter. Due to this similarity the accuracy was decreased to 80 percent. Seven letters have 70 percent of accuracy B, D, E, J, m, S and X. It is noticeable from figure.4.5 that B, D, and J have approximately similar shape and also need to perfect aligned with camera. The letters E, M, S and X have more similarity to each other and need the person try his/her best to show the letter clear in a distinguished. Finally letter N and T have the lowest accuracy (50 percent) because they have almost the

same shape and even it need to be careful and attention from individual to distinguish between them.

In Figures 4.8, 4.9, 4.10, 4.11 and 4.12 below, the output of the system is displayed using graphical user interface, the output for letter A,letter B, letter K, letter W and letter Z is shown as the examples.

Figure 4.8: The Output of Letter a

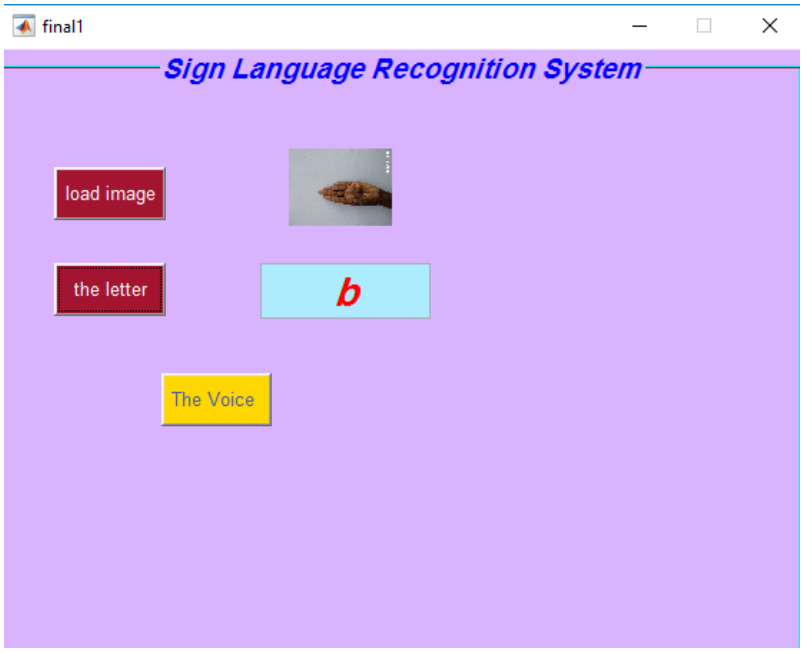

Figure 4.9: The Output of Letter b

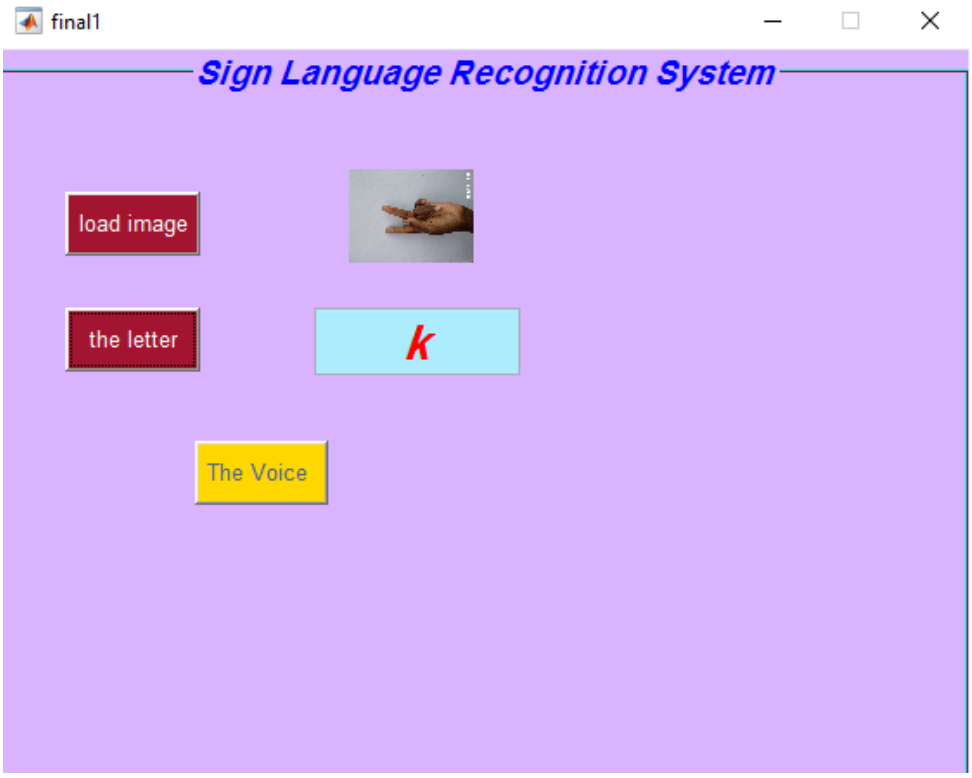

Figure 4.10: The Output of Letter k

Figure 4.11: The Output of Letter w

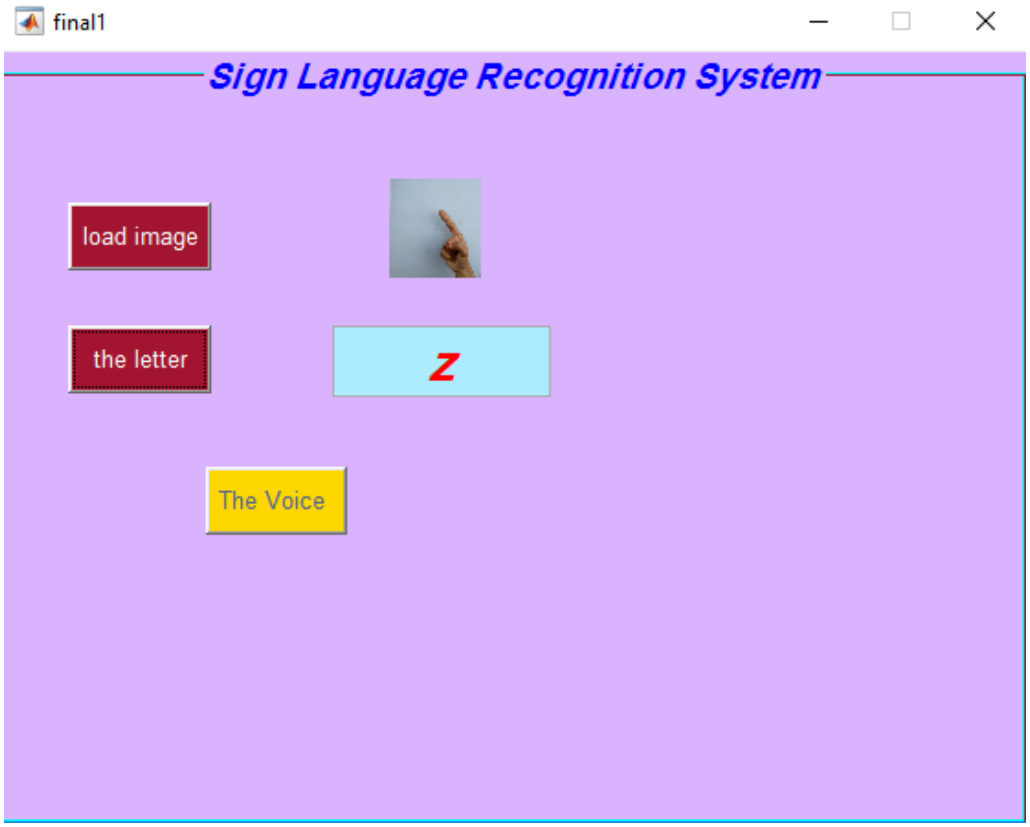

Figure 4.12: The Output of Letter z

Chapter Five
Conclusions and Recommendations

5.1 Conclusions

Communication with a normal person is always a challenging task for a dumb person. In this research a system called Sign language recognition is introduced which is an effective communication aid for a dumb person. The system uses the advanced technologies like color threshold, SURF algorithm, SVM classifier and bag of feature technique. It is convenient, comfortable and cheap; there is no need of a wearable to use the device. By using the text to speech conversion the system is extended to aid the deaf in communication. By converted system to GUI, the system is became more users friendly. The database is created with 260 images of hand gestures of American sign letters from different person with different colors, ages and posture, the data set was created by take ten photo for every letter, the accuracy of the system was increased with number of images in data set,However for technical constraints we cannot increase the number of images more than ten in each class, the dataset was partitioned , (90 percent) of the images for training and the remainder for testing because this present was given the most accuracy evaluating value . The system was tested for 10 people for 26 American sign letters, the accuracy is 84.6 percent. The factors were affected in the accuracy ; the people has different hands shape, their postures were not standard when the photo was taken, they did not align with camera perfectly, the resolution of camera phone is poor and the Similarity between some letters . The implementation of system output is done in MATLAB environment.

5.2 Recommendations

For enhancement this study, the database can be expanded the numbers of photo in dataset more than ten for every letter and take the photos for dataset from more expert people in dumb and deaf language. Also, for further future

enhancement, instead of using graphical user interface, the system can converted to application phone and connect between camera and application to make it easier and friendlier, and can convert to real time application by using video processing.

Bibliography

[1] S. S. SHINDE and D. R. AUTEE, "Real time hand gesture recognition and voice conversion system for deaf and dumb person based on image processing," *International Journal of Research Publications in Engineering and Technology [IJRPET]*, vol. 2, Sep 2016.

[2] J. Singha and K. Das, "Recognition of indian sign language in live video," *International Journal of Computer Applications (0975 – 8887)*, vol. 70, May 2013.

[3] A. K. Gautam and A. Kaushi, "American sign language recognition system using image processing method," *International Journal on Computer Science and Engineering (IJCSE)*, vol. 9, no. 7, Jul 2017.

[4] S. M. Mayuresh Keni and A. Marathe, "Sign language recognition system," *International Journal of Scientific and Engineering Research*, vol. 4, Dec 2013.

[5] M. D. Raut, P. Dhok, K. Machhale, and J. M. Hora, "A system for recognition of indian sign language for deaf people using otsu's algorithm," *International Research Journal of Engineering and Technology (IRJET)*, vol. 2, no. 01, 2015.

[6] F. N. Arko, N. Tabassum, T. R. Trisha, and F. Ahmed, "Bangla sign language interpretation using image processing," Ph.D. dissertation, BRAC University, 2017.

[7] WPClipart, "American Sign Language Alphabet," accessed 03-12-2018. [Online]. Available: https://wpclipart.com/sign_language/American_Sign_Language_Alphabet.png.html

[8] K. P. Kour and L. Mathew, "Sign language recognition using image processing," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 7, no. 8, 2017.

[9] I. Naoreen, "A systematic survey of skin detection algorithms, applications and issues," *Journal of Applied Environmental and Biological Sciences*, vol. 1, Sep 2014.

[10] S. O'Hara and B. A.Draper, "Introduction to the bag of features paradigm for image classification and retrieval," *arXiv preprint arXiv:1101.3354*, vol. 1, Jan 2011.

[11] S. P. Ebrahim Karami and M. Shehata, "Image matching using sift, surf, brief and orb: Performance comparison for distorted images," *arXiv preprint arXiv:1710.02726*, vol. 1, Oct 2017.

[12] H. Allen, Jonathan and D. M. Sharon;Klatt, *From Text to Speech: The MITalk system.* Cambridge University Press. ISBN 0-521-30641-8., 1987.

[13] B. T. M. Rubin, P., "An articulatory synthesizer for perceptual research," *Journal of the Acoustical Society of America*, no. 2, Aug 1981.

[14] S. R. W. O. J. P. H. J. Van Santen, Jan P. H., *Progress in Speech Synthesis.* Springer Science & Business Media. ISBN 0-387-94701-, 2013.

[15] S. Pramada, D. Saylee, N. Pranita, N. Samiksha, and M. Vaidya, "Intelligent sign language recognition using image processing," *IOSR Journal of Engineering (IOSRJEN)*, vol. 3, no. 2, pp. 45–51, 2013.

[16] S. R. U. P. M. Aswathy M, Heera Narayanan and J. J. UG, "Intelligent sign language recognition using image processing," *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, vol. 6, Mar 2017.

[17] A. P. Imunny, "Hybrid method for hand gesture recognition," *International Journal of Advance Research in Computer Science and Management Studies*, vol. 3, May 2015.

[18] M. U. Kakde and A. M. Rawate, "Hand gesture recognition system for deaf and dumb people using pca," *International Journal of Engineering Science and Computing*, vol. 6, July 2016.

[19] S. N. Sawant and M. Kumbhar, "Real time sign language recognition using pca," in *2014 IEEE International Conference on Advanced Communications, Control and Computing Technologies.* IEEE, 2014, pp. 1412–1415.

[20] A. S. Jagdish L. Raheja and Sadab, "Android based portable hand sign recognition system," *Dept. of Computer Science Truman State University USA*, 2015.

[21] S. A. E.-R. S. Mahmoud Zaki Abdo, Alaa Mahmoud Hamdy and E. M. Saad, "Arabic alphabet and numbers sign language recognition," *(IJACSA) International Journal of Advanced Computer Science and Applications*, vol. 6, no. 11, 2015.

[22] S. Đogić and G. Karli, "Sign language recognition using neural networks," *TEM Journal*, vol. 3, no. 4, 2014.

[23] A. El-Alfi, A. El-Gamal, and R. El-Adly, "Real time arabic sign language to arabic text & sound translation system," *Int. J. Eng*, vol. 3, no. 5, 2014.

[24] M. L. Jingzhong WANG, "A method of sign language gesture recognition based on contour feature," *Proceedings of the World Congress on Engineering and Computer Science*, vol. 1, 2014.

[25] S. R. J.J. and B. G. I. Ganesan, "Design and development of tamil sign alphabets recognition system using image processing to aid hearing impaired people," *Research Gate*, vol. 1, Sep 2011.

[26] B. Anand and M. P. K. Shah, "Face recognition using surf features and svm classifier," *International Journal of Electronics Engineering Research*, vol. 8, no. 1, pp. 1–8, 2016.

[27] R. Y. A. Ashi and A. A. Ameri, *Introduction to Graphical User Interface (GUI) MATLAB 6.5.* UAE University College Of Engineering Electrical Engineering Department.

Appendix

```
function varargout = final(varargin)
% FINAL MATLAB code for final.fig
%      FINAL, by itself, creates a new FINAL or raises the ...
    existing
%      singleton*.
%
%      H = FINAL returns the handle to a new FINAL or the ...
    handle to
%      the existing singleton*.
%
%      FINAL('CALLBACK',hObject,eventData,handles,...) ...
    calls the local
%      function named CALLBACK in FINAL.M with the given ...
    input arguments.
%
%      FINAL('Property','Value',...) creates a new FINAL or ...
    raises the
%      existing singleton*.  Starting from the left, ...
    property value pairs are
%      applied to the GUI before final_OpeningFcn gets ...
    called.  An
%      unrecognized property name or invalid value makes ...
    property application
%      stop.  All inputs are passed to final_OpeningFcn via ...
    varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI ...
    allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help final

% Last Modified by GUIDE v2.5 07-Mar-2019 10:42:44
```

```
26
27 % Begin initialization code - DO NOT EDIT
28 gui_Singleton = 1;
29 gui_State = struct('gui_Name',       mfilename, ...
30                    'gui_Singleton',  gui_Singleton, ...
31                    'gui_OpeningFcn', @final_OpeningFcn, ...
32                    'gui_OutputFcn',  @final_OutputFcn, ...
33                    'gui_LayoutFcn',  [] , ...
34                    'gui_Callback',   []);
35 if nargin && ischar(varargin{1})
36     gui_State.gui_Callback = str2func(varargin{1});
37 end
38
39 if nargout
40     [varargout{1:nargout}] = gui_mainfcn(gui_State, ...
          varargin{:});
41 else
42     gui_mainfcn(gui_State, varargin{:});
43 end
44 % End initialization code - DO NOT EDIT
45
46
47 % --- Executes just before final is made visible.
48 function final_OpeningFcn(hObject, eventdata, handles, ...
      varargin)
49 % This function has no output args, see OutputFcn.
50 % hObject    handle to figure
51 % eventdata  reserved - to be defined in a future version ...
      of MATLAB
52 % handles    structure with handles and user data (see GUIDATA)
53 % varargin   command line arguments to final (see VARARGIN)
54
55 % Choose default command line output for final
56 handles.output = hObject;
57
58 % Update handles structure
59 guidata(hObject, handles);
60
61 % UIWAIT makes final wait for user response (see UIRESUME)
62 % uiwait(handles.figure1);
63
64
```

```
65 % --- Outputs from this function are returned to the ...
       command line.
66 function varargout = final_OutputFcn(hObject, eventdata, ...
       handles)
67 % varargout  cell array for returning output args (see ...
       VARARGOUT);
68 % hObject    handle to figure
69 % eventdata  reserved - to be defined in a future version ...
       of MATLAB
70 % handles    structure with handles and user data (see GUIDATA)
71
72 % Get default command line output from handles structure
73 varargout{1} = handles.output;
74
75
76 % --- Executes on button press in pushbutton1.
77 function pushbutton1_Callback(hObject, eventdata, handles)
78 % hObject    handle to pushbutton1 (see GCBO)
79 % eventdata  reserved - to be defined in a future version ...
       of MATLAB
80 % handles    structure with handles and user data (see GUIDATA)
81 axes(handles.axes3);
82 global a;
83 [filename,pathname]=uigetfile({'".bmp';'".jpg';'"gif';'".*"'},'pick ...
       an Image Filee');
84 a=imread(filename);
85 imshow(a);
86
87 % --- Executes on button press in pushbutton2.
88 function pushbutton2_Callback(hObject, eventdata, handles)
89 % hObject    handle to pushbutton2 (see GCBO)
90 % eventdata  reserved - to be defined in a future version ...
       of MATLAB
91 % handles    structure with handles and user data (see GUIDATA)
92
93 % --- Executes on button press in pushbutton3.
94 function pushbutton3_Callback(hObject, eventdata, handles)
95 % hObject    handle to pushbutton3 (see GCBO)
96 % eventdata  reserved - to be defined in a future version ...
       of MATLAB
97 % handles    structure with handles and user data (see GUIDATA)
98 global a;
99 I = rgb2hsv(a);
```

```
100 channel1Min = 0.000;
101 channel1Max = 1.000;
102 channel2Min = 0.274;
103 channel2Max = 1.000;
104 channel3Min = 0.000;
105 channel3Max = 1.000;
106 BW = ( (I(:,:,1) ≥ channel1Min) | (I(:,:,1) ≤ channel1Max) ...
        ) & ...
107       (I(:,:,2) ≥ channel2Min ) & (I(:,:,2) ≤ channel2Max) & ...
108       (I(:,:,3) ≥ channel3Min ) & (I(:,:,3) ≤ channel3Max);
109 maskedRGBImage = a;
110 maskedRGBImage(repmat(~BW,[1 1 3])) = 0;
111 f=imresize(BW,[255 255]);
112 mystructhere=evalin('base','mystruct');
113 datahere=mystructhere.data;
114 [labelIdx, scores] = predict(datahere, f);
115 b=datahere.Labels(labelIdx);
116 set(handles.edit2,'string',b);
117
118
119
120 function edit1_Callback(hObject, eventdata, handles)
121 % hObject    handle to edit1 (see GCBO)
122 % eventdata  reserved - to be defined in a future version ...
        of MATLAB
123 % handles    structure with handles and user data (see GUIDATA)
124
125 % Hints: get(hObject,'String') returns contents of edit1 as ...
        text
126 %        str2double(get(hObject,'String')) returns contents ...
        of edit1 as a double
127
128
129 % --- Executes during object creation, after setting all ...
        properties.
130 function edit1_CreateFcn(hObject, eventdata, handles)
131 % hObject    handle to edit1 (see GCBO)
132 % eventdata  reserved - to be defined in a future version ...
        of MATLAB
133 % handles    empty - handles not created until after all ...
        CreateFcns called
134
```

```
135 % Hint: edit controls usually have a white background on ...
    Windows.
136 %       See ISPC and COMPUTER.
137 if ispc && isequal(get(hObject,'BackgroundColor'), ...
    get(0,'defaultUicontrolBackgroundColor'))
138     set(hObject,'BackgroundColor','white');
139 end
140
141
142
143 function edit2_Callback(hObject, eventdata, handles)
144 % hObject    handle to edit2 (see GCBO)
145 % eventdata  reserved - to be defined in a future version ...
    of MATLAB
146 % handles    structure with handles and user data (see GUIDATA)
147
148 % Hints: get(hObject,'String') returns contents of edit2 as ...
    text
149 %        str2double(get(hObject,'String')) returns contents ...
    of edit2 as a double
150
151
152 % --- Executes during object creation, after setting all ...
    properties.
153 function edit2_CreateFcn(hObject, eventdata, handles)
154 % hObject    handle to edit2 (see GCBO)
155 % eventdata  reserved - to be defined in a future version ...
    of MATLAB
156 % handles    empty - handles not created until after all ...
    CreateFcns called
157
158 % Hint: edit controls usually have a white background on ...
    Windows.
159 %       See ISPC and COMPUTER.
160 if ispc && isequal(get(hObject,'BackgroundColor'), ...
    get(0,'defaultUicontrolBackgroundColor'))
161     set(hObject,'BackgroundColor','white');
162 end
163
164
165 % --- Executes on button press in pushbutton5.
166 function pushbutton5_Callback(hObject, eventdata, handles)
167 % hObject    handle to pushbutton5 (see GCBO)
```

```
168 % eventdata  reserved - to be defined in a future version ...
      of MATLAB
169 % handles    structure with handles and user data (see GUIDATA)
170 global a;
171 numericVector = cell2mat(b);
172 NET.addAssembly('system.speech');
173 mySpeaker=System.Speech.Synthesis.SpeechSynthesizer;
174 mySpeaker.Rate=0;
175 mySpeaker.Volume=100;
176 Speak(mySpeaker,numericVector)
```