



**PASSWORD AUTHENTICATION USING
BIOMETRIC FUNCTION
(VOICE / SPEAKER RECOGNITION)**

Perpustakaan SKTM

**MOHD SHAHMAN BIN SAMSUL AMBIA
WEK 990261**

**FAKULTI SAINS KOMPUTER
DAN
TEKNOLOGI MAKLUMAT
UNIVERSITI MALAYA
SESI 2002/2003**

ACKNOWLEDGEMENT

A lot of effort has been put in towards making this proposal and a lot of help has been received. Here I would like to express my acknowledgement to the people that have helped me.

First, I would like to thank my supervisor Dr. Syed Malek Fakar Duani. He has been a great source of inspiration and has provided the right balance of help, suggestions, criticism, and freedom. I also would like to thank Mr. Woo Chaw Seng, as a moderator for his opinion and suggestion in making this project.

Finally, I would also like to express my gratitude to my family and friends that supported me in making this project.

TABLE OF CONTENTS

Acknowledgement	ii
Table of Contents	iii
List of Tables	viii
List of Figures	ix
Abstract	x
 Chapter 1 : Introduction	 1
1.1 Overview	1
1.2 Project Objective	1
1.3 Project Scope and Limitations	2
1.3.1 Scope	2
1.3.2 Limitations	2
1.4 Project Planning	3
 Chapter 2 : Literature Review	 6
2.1 Biometrics	6
2.1.1 Introductions to Biometrics	6
2.1.2 Biometrics Methods	7
2.1.2.1 Fingerprint Identification	7
2.1.2.2 Hand Geometry	8
2.1.2.3 Eye Scanning	8
2.1.2.4 Face Recognition	9
2.1.2.5 Voice Verification	9
2.1.2.6 Signature Recognition	10
2.1.2.7 Keystroke Dynamics	11
2.1.2.8 DNA	11
2.1.3 Future Applications	11
2.2 Speaker Recognition	12
2.2.1 Speaker verification and speaker identification	12
2.2.2 Working process of speaker verification	13
2.2.3 Speaker verification approaches	15

2.2.3.3 User customized password speaker verification	19
2.2.3.4 Text-independent speaker verification	20
2.3 Speech Signal Analysis	22
2.3.1 Analogue-to-digital (ADC) conversion	22
2.3.2 Preprocessing	23
2.3.2.1 Speech enhancement	23
2.3.2.2 Speech framing	24
2.3.2.3 Windowing	24
2.3.2.4 Pre-emphasis	25
2.3.2.5 End-Point Detection	26
2.3.3 Features calculation	27
2.3.3.1 Mel-Frequency Cepstral Coefficients (MFCC)	29
2.3.4 Feature analysis	33
2.3.4.1 Feature Normalization	33
2.3.4.2 Feature Comparison	34
2.4 Signal Classification	35
2.4.1 Artificial Neural Network	30
2.4.1.1 Introduction	36
2.4.1.2 Biological Neural Network & Artificial Neural Network	36
2.4.1.3 Neural Network Types	39
2.4.2 Statistical/Stochastic Method	42
2.4.2.1 Hidden Markov Models	42
2.4.3 Template Methods	
2.4.3.1 Vector Quantizations	43
Chapter 3 : Methodology	47
3.1 Introduction	47
3.2 System Development Life Cycle (SDLC)	47
3.2.1 Requirement Analysis	48
3.2.2 System & Program Design	48
3.2.3 Coding	48
3.2.4 Testing	48

3.2.5 Operation & Maintenance	49
Chapter 4 : System Analysis	50
4.1 Functional Requirements	50
4.1.1 Voice Acquisition	50
4.1.2 Speech Signal Preprocessing	50
4.1.3 Feature Extraction	50
4.1.4 Feature Clustering	51
4.1.5 Verification	51
4.2 Non-functional Requirements	51
4.2.1 System Reliability	52
4.2.2 User-friendly Interface	52
4.2.3 Response Time	52
4.2.4 Robustness	52
4.3 Technical Requirements – Hardware	52
4.4 Technical Requirements – Software	52
4.4.1 Development Tools	52
4.4.2 Operating System	53
Chapter 5 : System Design & Architecture	54
5.1 System Design	54
5.1.1 Voice Acquisition	54
5.1.2 Speech Signal Pre-processing & Feature Extraction	54
5.1.3 Clustering And Pattern Matching	54
5.1.4 Threshold Creation	54
5.1.5 Verification	55
5.2 System Flow Diagram	56
5.2.1 Enrollment Module	56
5.2.2 Verification Module	57
5.3 System Architecture	58
Chapter 6 : System Implementation	59
6.1 Platform	59
6.2 Program Usage	59

6.3	High Level Organization	
6.4	Functional Details	59
6.4.1	Signal Pre-Processing	59
6.4.1.1	Endpoint Detection	60
6.4.2	MFCC	60
6.4.3	Vector Quantization (VQ)	61
6.4.4	Threshold Creation	61
6.4.5	Decision	62
Chapter 7 : System Testing		63
7.1	Integration Test	63
7.2	System Test	63
7.2.1	Problems encountered during system testing	64
7.3	Performance Test	64
7.3.1	Optimization Tests	64
7.3.2	Execution Time Test	65
7.3.3	Disk Space Usage	65
Chapter 8 : System Discussion & Recommendations		66
8.1	System Discussion	66
8.1.1	Enrollment Module	66
8.1.2	Threshold Module	66
8.1.3	Verification Module	67
8.1.4	Microphone Issues	67
8.2	Alternative Options	68
8.2.1	Feature Extraction Alternatives	68
8.2.2	Verification Alternatives	68
8.3	Recommendations	68
8.3.1	Threshold Generation Using Multiple Verification Sessions	68
8.3.2	Weighting The Code Vectors	69
8.3.3	Codebook Adaptation Over Time	69
8.3.4	Signal Normalization	69
8.3.5	Verification Threshold Options	69

Conclusion	70
List Of References	71
Appendix A: GUI & Manual	73
Appendix B: Test Results	78
Appendix C: Call Structures	79

LIST OF TABLES

TABLE	TITLE	PAGE
1	Table 1.1 : 1 st semester project planning	3
2	Table 1.2 : 2 nd semester project planning	4
3	Table 2.1: Overview of commercially available products	21
4	Table 2.3: Multi-Layer-Perceptron	40
5	Table 2.4: Backpropagation net	41
6	Table 6.1: Default Mel-Cepstrum Parameters	61
7	Table 7.1. Optimal Parameter Values	65

LIST OF FIGURES

FIGURES	TITLE	PAGE
1	Figure 1.1 : Current activities Gantt Chart	4
2	Figure 1.2 : 2 nd Semester activities Gantt Chart	5
3	Figure 2.1: Basic structure of speaker verification	15
4	Figure 2.2: Time domain response of some common window functions	25
5	Figure 2.3: Block diagram of MFCC processor.	29
6	Figure 2.4: MFCC Filterbank	32
7	Figure 2.5: Structure of a neural cell in the human brain	37
8	Figure 2.6: Structure of a neuron in a neural net	37
9	Figure 2.7: Neural net with three neuron layers.	39
10	Figure 2.8: A fully connected three state HMM	42
11	Figure 2.9: A left-to-right HMM model	43
12	Figure 2.10: Conceptual diagram illustrating vector quantization codebook formation	44
13	Figure 2.10: Flow diagram of the LBG algorithm	46
14	Figure 3.1: The V-Model.	47
15	Figure 5.1: Flow chart of enrollment module	56
16	Figure 5.2: Flow chart of verification module.	57
17	Figure 5.3: The proposed Speaker Verification system architecture flow	58
18	Figure 6.1: Example of speech period extraction	60

ABSTRACT

This project's main purpose is to apply the use of a biometric function on a password authentication system. A pattern recognition based on the template matching method was used on the task of speaker verification. In the proposal of the system, I opted for the use of dual classification that uses Multi Layer Perceptron(MLP) and the stochastic models of Hidden Markov Models(HMM). However, after careful considerations and lots of trials and errors, the method that was chosen is the Vector Quantization(VQ) method. This is due to the fact that VQ is less computationally complex when compared to stochastic models such as the HMM which is much more computationally expensive.

The product of this project can be considered a success as it has fulfilled all of its requirements and the implementation of the VQ on a speaker verification system can successfully be done.

CHAPTER 1

INTRODUCTION

1.1 Overview

Biometrics refers to the automatic identification of a person based on his/her physiological or behavioral characteristics. This method of identification is preferred over traditional methods involving passwords and PIN numbers for various reasons: (i) the person to be identified is required to be physically present at the point-of-identification; (ii) identification based on biometric techniques obviates the need to remember a password or carry a token.

Speaker recognition collectively describes the tasks of extracting or verifying the identity of the speaker. In speaker identification, the task is to use a speech sample to select the identity of the person that produced the speech from among a population of speakers. In speaker verification, the task is to use a speech sample to test whether a person who claims to have produced the speech did in fact do so.

Using a speaker recognition system is usually a two-step process. The user first enrolls by providing the system with one or more representative samples of his or her speech. These training samples are then used by the system to train a model for the user. In the second step the user provides a test sample that is used by the system to test the similarity of the speech to the model(s) of the user(s) and provide the required service.

1.2 Project Objectives

The main aim of the Speaker Verification System project is to enhance securities by using voice biometrics as an alternative to normal password

authentication systems. Based on this aim, the objectives of this system is to provide

:

- A speaker verification system which uses a classification method.
- Allowing of user-customized passphrase which allows user to choose the passphrase that they want without the constraint of meaning and language.
- A speaker verification system that is robust to noise by using signal preprocessing techniques to get a better signal.

1.3 Project Scope & Limitations

1.3.1 Scope

The main scope of this project is to decide and implement a classification method in a speaker verification application. This involves the development (i.e. coding) of the chosen method(s) and testing the system for its feasibility in commercial implementation.

1.3.2 Limitations

Limitations of this project among others are :

- Speaker verification is not the most effective method of biometric authentication since the search of a distinctive feature that truly belongs to specific person has not been successful.
- Noise in signals are impossible to avoid since there are too many forms of noise around. Even though noise reduction preprocessing will be implemented to reduce the level of noise, it may still affect the results of the classification process thus reducing its efficiency.

1.4 Project Planning

Scheduling and planning of the activities that needs to be done for the duration of the project development process are prepared early on to assure that the project can be developed fully in the time constraint given. The duration of development of this project is 2 semesters (about 30 weeks).

This project is divided into two (2) phase :

- Project Proposal (“*Latihan Ilmiah 1*”)
- System Development (“*Latihan Ilmiah 2*”)

Below is the project implementation table :

Semester I (2002/2003)

Table 1.1 : 1st semester project planning

Week	Description
1	Title listing
2	Title choosing and problem domain
3	Information mining from written references or the internet
4	Information searching and collection
5	Information searching and collection
6	Information searching and collection
7	Information searching and collection
8	Information searching and collection
9	Information searching and collection
10	Information searching and collection, and preparation of project proposal.
11	Information searching and collection, and preparation of project proposal.
12	Project proposal hand-in

Semester II (2002/2003)

Table 1.2 : 2nd semester project planning

Week	Description
1	Knowledge Acquisition, design and coding
2	Designing and coding
3	Designing and coding
4	Coding
5	Coding
6	Coding
7	Coding
8	Coding and testing
9	Coding and testing
10	Coding and testing
11	Coding, testing and maintenance
12	Project documentation and user manual preparation
13	Project documentation and user manual preparation

On the next page are the Gantt Chart of current activities (Figure 1.1) and the Gantt Chart for 2nd semester activities (Figure 2.2).

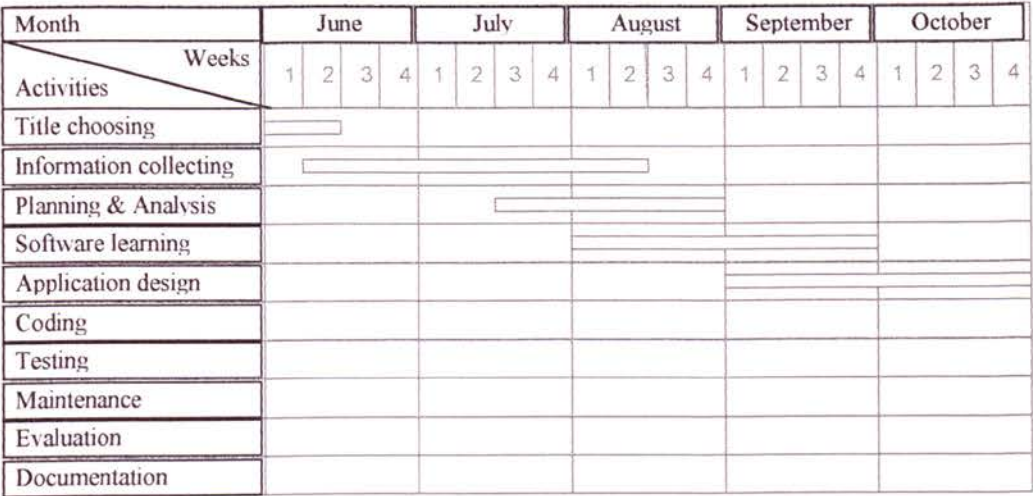


Figure 1.1 : Current activities Gantt Chart

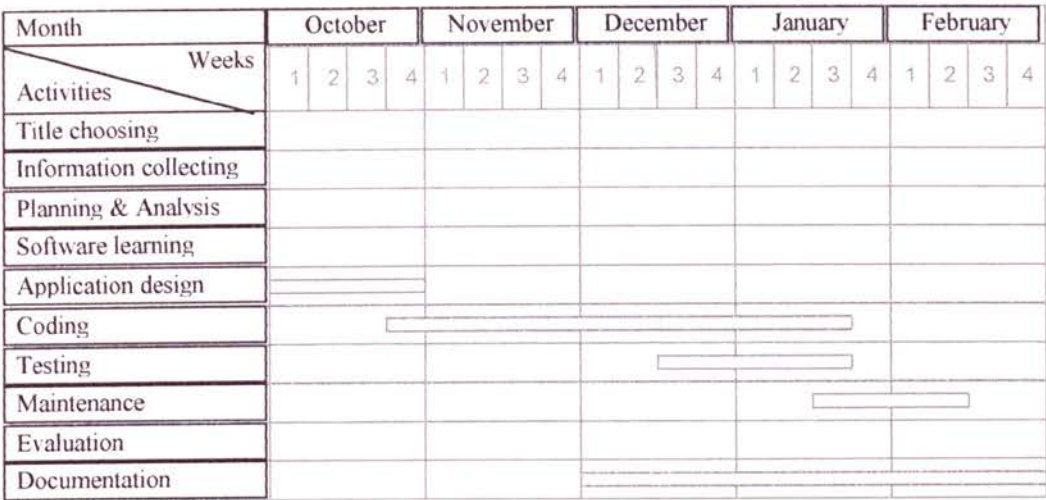


Figure 1.2 : 2nd Semester activities Gantt Chart

CHAPTER 2

LITERATURE REVIEW

The purpose of this literature review is to get a better understanding on the underlying technology and the methods used to develop this project. The information here are mostly gathered from the internet and from written books and papers.

2.1 Biometrics

2.1.1 Introduction to Biometrics

Biometrics is an old Greek word for a very new concept. "Bio," meaning life, and "Metric," the measure of, so Biometrics is in essence, the measure of life. Biometrics is an emerging technology for automatically identifying individuals using their distinct physical or behavioral characteristics. Types of Biometrics can be any unique human characteristics, such as fingerprint, voice, face, iris, retina, palm, signature, wrist vein, and hand geometry. Biometrics provides a better solution for the increased security requirements of our information society than current identification methods (passwords, PIN numbers and magnetic strip cards with a PIN number) for various reasons: the person to be identified must be physically present at the point of identification; identification based on biometric techniques obviates the need to remember a password (or write it on a yellow sticky note), PIN or carry a token. Using biometric systems to identify the user of a computer, ATM, cellular phones and even credit card purchases will reduce fraud and unauthorized access. This could save the economy billions of dollars.

There are issues that must be defined when designing a practical system. Two initial issues are determining how the person is going to be identified, verification or identification. Verification involves confirming or denying a person's claimed identity. In verification, the biometric system will verify whether or not an individual's biometric sample matches a previously enrolled data. Identification establishes a person's identity. In identification, a biometric sample is compared against a database of stored users. Both approaches have complexities and could probably be solved best by a certain biometric system.

2.1.2 Biometrics Methods

A number of applications using various biometrics functions are in use or in research currently. Here are stated the types of available biometrics methods.

2.1.2.1 Fingerprint identification

Fingerprint identification is the most commonly recognized and most widely applied form of Biometric technology. Fingerprint ID is based upon the fact that a person's fingerprint is completely unique to the individual. A fingerprint is made of a series of ridges and furrows on the surface of the finger. The uniqueness of a fingerprint can be determined by the pattern of ridges and furrows as well as the minutiae points.

In the past, fingerprints were recorded by the application of ink to the finger which was then pressed to the paper to give an impression. More recently, it has become possible to scan a person's fingerprint into virtual storage in a computer with the aid of laser technology. In order to prove identification, a person's fingerprint will be scanned again in the future by a similar device, and a match of print to name is verified through information systems. Techniques such as this are even currently

being used in applications as commonplace as automatic teller machines and security locks.

2.1.2.2 Hand geometry

Hand geometry is even older than digital fingerprinting; it was first used for security purposes on Wall Street more than 20 years ago. We're already conditioned to offer a hand in friendship or to seal an agreement, which may be why hand scanners have gained wide acceptance in office buildings, factories, and other corporate environments.

Hand geometry is based on the fact that virtually every person's hand is shaped differently and that the shape of a person's hand (after a certain age) does not significantly change. When the user places a hand on the hand reader, a three-dimensional image of the hand is captured. Then, the shape and length of the fingers and knuckles are measured. Depending on the data used to identify a person, hand reading technologies generally fall into one of three categories - application to the palm, the pattern of veins in the hand and the geometrical analysis of fingers.

2.1.2.3 Eye scanning

Biometrics which analyze the complex and unique characteristics of the eye can be divided into two different fields: iris biometrics and retina biometrics. The iris is the colored band of tissue that surrounds the pupil of the eye. An iris recognition system uses a video camera to capture the sample while the software compares the resulting data against stored templates.

The retina is the layer of blood vessels at the back of the eye. Retina scans are performed by directing a low-intensity infrared light to capture the unique retina

characteristics. An area known as the fovea, situated at the center of the retina, is scanned and the unique pattern of the blood vessels is captured. Retina biometrics is considered to be the best biometric performers. However, despite its accuracy, this technique is often thought to be inconvenient and intrusive. And so, it is difficult to gain general acceptance by the end user. The retinal scanner requires an individual to stand still while it is reading the retinal information. Eye and retinal scanner are ineffectual with the blind and those who have cataracts.

2.1.2.4 Face recognition

Face recognition systems identify an individual by analyzing the unique shape, pattern and positioning of facial features. There are essentially two methods of processing the data: video and thermal imaging. Standard video techniques are based on the facial image captured by a video camera. Thermal imaging techniques analyze the heat-generated pattern of blood vessels underneath the skin.

The attraction of this biometric system is that it is able to operate 'hands free', limiting the amount of man-machine interaction. However, this system is highly unreliable and expensive. For example, it will not distinguish twins or triplets, not recognize the user after a haircut, and not recognize a person who changes from wearing and not wearing glasses.

As concerns face recognition many approaches have been proposed in the literature, and several researchers are studying this problem. Principal component analysis, elastic graph matching, neural networks, and distortion-tolerant template matching are only a few of the proposed techniques.

2.1.2.5 Voice verification

Voice verification is the science of verifying a person's identity on the basis of their voice characteristics. Unique features of a person's voice are digitized and compared with the individual's pre-recorded "voiceprint" sample stored in the database for identity verification. It is different from speech recognition because the technology does not recognize the spoken word itself. Rather, it recognizes the speaker of the words by analyzing unique speech characteristics, such as the frequency between phonetics.

Speaker recognition technology makes it possible for a speaker's voice to control access to restricted services, for example, phone access to banking, database services, shopping or voice mail, and access to secure equipment.

While speaker recognition is convenient, it is not as reliable due to the risks of impersonation, remote access and bad accuracy. A person with a cold or laryngitis may have problems using a speaker recognition system due to false rejection.

2.1.2.6 Signature recognition

The user signs his signature on a digitized graphics tablet. Signature dynamics, such as speed, relative speed, stroke order, stroke count and pressure are analyzed. The key in signature dynamics is to differentiate between the parts of the signature that are habitual and those that vary with almost every signing.

The use of a signature itself has been widely accepted. Still, problems with signature recognition lie in the means of obtaining the measurements used in the recognition process and the repeatability of the signature. The DSV (Dynamic Signature Verification) system is engineered in a way to adapt to variances.

Nevertheless, without lowering the acceptance rate, it cannot consistently measure the dynamics of a signature.

2.1.2.7 Keystroke dynamics

Keystroke dynamics analyze the way a person types. It is a very new technology to the biometrics arena. Users enroll by typing the same word or words a number of times. Verification is based on the concept that the rhythm with which one types is distinctive.

2.1.2.8 DNA

The term "DNA" is actually short for Dioxyribo Nucleic Acid. DNA is found in every cell of every creature, and it contains the information for carrying out the activities of the cell. Since every person's DNA structure is completely unique, DNA analysis is a very accurate way of proving identification. Due to the extensive testing and advanced technology required, it is not the most cost efficient Biometric science, but when a positive identification is needed it is the most reliable.

Its use, however, is quite limited to medical or forensics application. The use of DNA as an extension to password authentication is downright impossible with the current technologies.

2.1.3 Future applications

Biometrics is a rapidly evolving technology which is being widely used in forensics such as criminal identification and prison security, and has the potential to be used in a large range of civilian application areas. Biometrics can be used to prevent unauthorized access to ATMs, cellular phones, smart cards, desktop PCs, workstations, and computer networks. It can be used during transactions conducted

via telephone and internet (electronic commerce and electronic banking). In automobiles, biometrics can replace keys with key-less entry devices.

2.2 Speaker Recognition

Speaker recognition has many potential applications, which include: secured use of access cards (e.g., calling and credit credits), access control to databases (e.g., telephone and banking applications), access control to facilities, electronic commerce, information and reservation services, remote access to computer networks. Speaker recognition can also be used in military and forensic applications.

2.2.1 Speaker verification and speaker identification

Speaker recognition usually falls into two categories: speaker identification and speaker verification. These two divisions are often done simultaneously and both of them rely on voiceprints – the unique characteristics of every person's voice; however, they are used in different ways and for different purposes.

Speaker recognition deals with the tasks of extracting or verifying the identity of the speaker. When using speaker recognition systems, if the customer says, "it is Ali", then the system only have tell whether he is "Ali" or not, and the task is a speaker verification problem; whereas if the customer says "it is me", the system will have to find out who the speaker is, thus it has to perform a speaker identification. In short, in speaker identification, the task is to use a speech sample to tell the identity of the person that produced the speech from a population of speakers. This task involve the calculation of N possibilities, where $N > 1$ is the population of speakers. In speaker verification, the task is to use a speech sample to tell whether a person is who he claims to be. The number of possible choices in speaker identification is the

number of speakers in the population, whereas in speaker verification the outcome of the recognition process is limited to one of two choices: accept or reject.

Depending on whether the speaker population is known to the system or not, speaker identification can be further divided into closed set identification and open set identification. Closed set speaker identification is the task where every speaker in the population is known to the system before using the system. Open set identification is the task where some speakers in the population are unknown to the system at the time of use and thus should be rejected. From this sense, open set identification is a combination of closed set identification and speaker verification, which means we may have to verify if the speaker has enrolled in to the system and tell who he/she is if the speaker is registered in the system.

For this project only the speaker verification method will be covered so the next few sections will be about speaker verification only.

2.2.2 Working process of speaker verification

Using a speaker recognition system is usually a two-step process. The user first enrolls by providing the system (usually computer) with some representative samples of his/her speech, called training samples. These training samples are then used by the system to construct a model/template for the user. When to provide these samples, how to produce these samples, and how many samples are needed depend on the specific systems. For example, whether the system is text-dependent or text-independent will determine the format of the speech samples. In the second step the user provides a test sample that is used by the system to test the similarity of the speech to the model(s) of the user(s) and provide the required service. In this second step the speaker associated with the model that is being tested is termed the target speaker or claimant. Again, how this is processed depends on the system used.

In more detail, a speaker recognition process often involves the following stages that are common to both speaker verification and speaker identification.

- a) Data acquisition, where the speech produced by the speaker is converted from a sound pressure waveform into an electrical signal using a transducer. This acoustic signal is digitized and sampled at a suitable rate. A microphone, phone or other such kind of devices will do the job; there is no special requirement.
- b) Signal processing and feature extraction, where salient parameters conveying speaker identity are extracted from the acoustic speech signal. Signal processing includes noise cancellation. Feature extraction is to collect the speaker specific info from the acquired sampled, it is based on the existing knowledge base of the speech process -- such as models of the articulator and auditory systems, theory of linguistics and phonetics, perceptual cues used by listeners, transmission process, and application specific requirements.
- c) Similarity measure. In terms of speaker verification, this measure is done between the information retrieved from the speech of the current speaker and a previously constructed model representing the person the speaker claims to be. The model training forms a major component of the speaker verification system. It determines storage cost and computation and dictates accuracy of the similarity measure.
- d) Decision-making – the final stage. For speaker verification, in this stage, the similarity measure is compared to a threshold and decides whether to accept or reject the claimed identity of the speaker (for speaker verification). For example, if the model of the claimed speaker is deemed to represent the information retrieved from the acoustic signal just acquired accurately, i.e.

the similarity measure is greater than the threshold, then the decision is to accept the claim made by the speaker.

Fig. 2.1 shows a typical speaker verification process. It begins by providing the system with a claim of identity, such as entering an account number or ID code, presenting a credit card or ATM card, or allowing the system to access the ID of a user-linked input device, such as a specific cellular telephone. Typically, a person enters the ID manually using a telephone keypad or keyboard. If the system uses speech recognition in conjunction with speaker verification, verbal input of an ID code may be requested.

The system uses the ID to retrieve the reference voiceprint for the person authorized to use the ID. It then requests a sample of speech from the claimant. The newly input speech is converted into a voiceprint and compared to the reference voiceprint template. The results of the comparison are quantified and compared to an acceptance/rejection threshold to determine whether the two voiceprints are similar enough for the system to accept the identity claim.

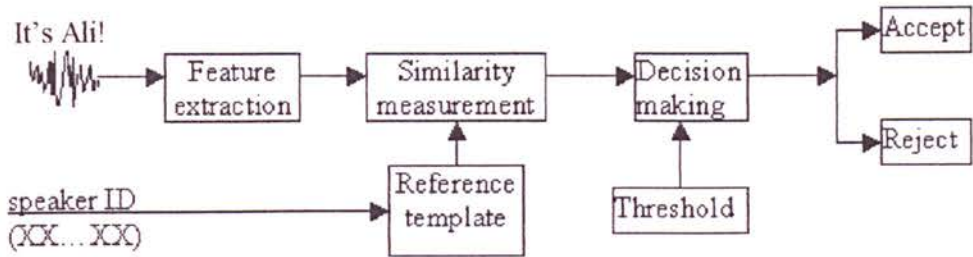


Figure 2.1: Basic structure of speaker verification

2.2.3 Speaker verification approaches

Speaker recognition can be performed based on text-dependent or text-independent utterances, depending on whether or not the recognition process require to speak the same message during both training and testing. According to whether the speech is text-dependent or text-independent and how it is carried out, there are usually 4 sorts of speaker verification approaches: text-dependent speaker verification, text-independent speaker verification, text-prompted speaker verification and user-customized password speaker verification. What they are and how they work will be briefly discussed in the following sections.

2.2.3.1 Text-dependent speaker verification

In speaker verification, when the person is constrained to speak the same sentence during both training and testing processes the task is text-dependent speaker verification. There are many ways to choose the sentence, which is often termed verification phrase or passphrase. For example, the verification phrase may be a unique password or a fixed string of digits. In this case, model of each speaker contains both speaker specific characteristics as well as characteristics of lexical content of the utterance. The text-dependent methods are usually based on template matching techniques in which the time axis of an input speech sample is aligned with each reference template or reference model of registered speakers, and the computed similarity between them is accumulated from the beginning to the end of the utterance, then the final similarity is then compared with a preset threshold to determined the output. The structure of text-dependent verification methods is therefore rather simple.

Text-dependent system requests a password. This system provides “strong authentication”, which requires the use of at least two different kinds of security. In the example, to be accepted by the system, the person must have the correct voice and also know the proper password.

Dynamic time warping (DTW) and Hidden Markov Model (HMM) are two of the principal approaches in text-dependent speaker verification. In DTW methods, each passphrase is represented by a sequence of small number acoustic templates, during recognition a core of a new text passphrase is computed using dynamic time warping algorithm against the templates. It is rather simple and takes less computational resources to store.

HMM methods usually achieve better recognition performance than DTW methods. In this case, each passphrase is modeled by a HMM, the parameters of the HMM's are trained from repetitions of the passphrase, when the used HMM's are large, the amount of training material can be a problem in the practical applications.

Since text-dependent exploits directly the voice individuality associated with each phoneme or syllable, it generally achieves higher recognition performance than the text-independent method. While text-dependent verification potentially requires only a small amount of speech it requires the user to faithfully produce the required text. Thus it requires a cooperative user and a structured interaction between the user and system. Companies such as SpeakEZ (SpeakEZ, Inc.), Watson (AT&T Corp) provide text-dependent speaker verification technologies.

2.2.3.2 Text-prompted speaker verification

Since verification is based on both the speaker characteristics and the lexical content of a secret password, text-dependent speaker verification systems are generally more robust than text-independent systems. However, both of them are susceptible to fraud

because the voice of the speaker could be captured, recorded, and reproduced. In the case of a text-dependent system, even the password could be captured. Speaker verification systems based on prompted text have thus been developed to reduce this sort of risks.

In this method, for each access, a recorded or synthetic prompt will ask the user to pronounce a different random sentence. The underlying lexicon could either be very large, or very small - limited to the 10 digits, which would then be used to generate random digit strings. The advantage of such an approach is that impostors cannot predict the prompted sentence. Consequently, prerecorded utterances from the customer will be of no use to the impostor.

During each access, the system prompts the user with a different sentence and a speaker-independent speech recognition system will be used prior to verification to make sure the user actually repeated what he/she was prompted to say. When a speaker is rejected because of the recognized text of the utterance differs from the prompted text, he/she can still be prompted with an additional sentence. Since the new sentence will be different, the acoustic vector sequence will not be too correlated with the previous one, which will improve the quality of the system since more uncorrelated evidence is collected. For example, the user maybe randomly prompted for two pass phrases. If both utterances provided pass the biometrics test, the user is recognized. If one of the two pass phrases is rejected, the system prompts for the third pass phrase that, if accepted, allows the user to further access. If two of the three questions are missed, the user is denied access. This strategy cannot be expected to work as well in a text dependent system, since the repeated sentence in text dependent system would have the same lexical content as the original one.

In this approach, speech recognition based on phonetic HMMs is used before verification. The models are defined to cover the lexicon, and are independently trained for each user. The most challenging part with approach is that there is usually the lack of enough data to train the Hidden Markov Models. During verification, the system knows the prompted sentence and can build the associated HMM model which is then used to validate the utterance by computing the confidence level associated with the acoustic vector sequence, and perform speaker verification.

2.2.3.3 User-customized password speaker verification

In commercial services, it is often desirable that the user can choose his/her own password, which is then used for verification. This method used to provide speaker verification in such a way is referred to as user-customized password SV. It enables better user-friendliness, which is a very common concern and considered as an essential functionality by some service providers for implementing speaker verification technology. This method also enables increased security as fraudulent access requires prior knowledge of the client's password and the user can change his/her password whenever he/her wants to.

Compared with the text-prompted speaker verification, the main difference is that in text-prompted speaker verification, the system knows the text of the training utterances in advance. Whereas in the case of user-customized password speaker verification, the password is a priori unknown to the system, so it has to first automatically infer the lexical characteristics of the customized password, simply on the basis of a few pronunciations of that password. The enrollment generally consist the following two steps: 1. HMM inference: Infer the HMM model associated with the used-specific password from a few repetitions (usually 2 or 3) of the password. 2.

HMM adaptation: the parameters of this model must be adapted to the characteristics of the customer's voice.

Another difficulty related to this approach is the normalization of the score, usually using a "world model", i.e., a model of the way the rival speakers (impostors) would pronounce this password. So, the problem is also to infer speaker-independent model of the password from the single-speaker set of examples.

2.2.3.4 Text-independent speaker verification

In text-independent speaker verification, the person is not constrained to speak the same sentence during training and testing; the system accepts any spoken input. This makes it possible to design an unobtrusive, even invisible verification application. This is required in many applications where the user may be uncooperative or applications where speaker recognition occurs as a secondary process unknown to the speaker. For example, a forensic application may require verifying the identity of a speaker based on speech from a recorded telephone conversation and the speaker may not actually be aware of this process.

Different methods are investigated to this problem, among them are: Long-Term-Statistics method, which are based on simple statistics such as the long-term mean and variance that are calculated over a series of utterances; VQ based methods in which spectral characteristics of each speaker are modeled by one or more codebooks that are obtained by clustering training data of each speaker; Gaussian Mixture Model (GMM) that is proved to yield good performance in text-independent speaker verification; Artificial neural network in which each speaker has his/her own neural network.

Text-independent technology is much more difficult to implement than text-dependent and text-promoted technology. In this approach, the system only makes

use of the speaker's voice characteristics, because of the higher acoustic-phonetic variability of text-independent input, more training material is necessary to reliably characterize (model) a speaker than with text-dependent methods. This method is also more sensitive to the acoustic quality of the input. Another potential concern arises from the privacy issue, since the technology can be used without the subject's knowledge. Companies that provide Text-independent technology includes VoiceID Systems, Inc.

Below is an overview of the products that uses voice biometrics in the current market

Product	Overview	Company
BioID [4]	BioID uses face, voice and lip movement recognition to identify a person for security and identification purposes	BioID
SpeakEZ Voice Print Speaker Verification [5]	The technology compares a digitized sample of a person's voice with a stored "voice print" of that individual's voice for verification.	T-NETIX Incorporated
VeriVoice Security Lock [6]	The VeriVoice Security Lock is a patented voice verification technology that provides a fast, highly accurate, and customizable solution for identification of enrolled users.	VeriVoice Inc.
VoiceCheck [7]	It handles all the enrollment, voice checking (verification), and database management needed to incorporate a voice verification solution	Veritel Corporation
CAS Guard [8]	Biometric authentication. Admin logon requires administrator to biometrically logon (using face, finger, or voice) before making changes and updates to the system.	Keyware Technologies
Speak N Set	Designed for use with PC files. You can use a human voice as a password, as unique as a	Veritel Corporation

[9]	human voice as a password, as unique as a fingerprint, as natural as saying hello.	Corporation
-----	--	-------------

Table 2.1: Overview of commercially available products

2.3 Speech Signal Analysis

The process of speech analysis involves converting the recorded analogue speech signal into feature vectors that can be applied in the speaker verification application.

This generally involves four steps:

- Analogue-to-digital conversion.
- Preprocessing.
- Feature calculation.

2.3.1 Analogue-to-digital (ADC) conversion

A digital computer cannot operate on the pure acoustic waves (as transmitted through air or other mediums) or its electronic analogue form. The analogue speech signal must first be converted to a digital speech signal that can be discretely manipulated by a digital computer. This is done by an analogue to digital converter (ADC), which is usually a specialized component inside either a digital signal processing (DSP) card or a sound card in a digital computer. Since noise inflict such a huge penalty on recognition systems, the quality of the ADC can contribute significantly to the success of the recognition system.

ADC can be subdivided into three steps:

- Sampling.
- Quantization.
- Coding.

Sampling is the process of taking a sample from the continuous analogue speech signal at regular intervals, and which is denoted the sampling interval. According to the sampling theorem, the analogue signal should be sampled at more than twice the highest operating frequency to avoid aliasing. Depending on the operating environment, a tradeoff between the sampling frequency and the quantity of data can be made. CD quality audio data is sampled at 44.1 kHz, high quality speech usually between 16 and 20 kHz, while low quality telephone transmitted speech is sampled at around 8 kHz (because of the low bandwidth that telephone channels offer).

Quantization is the process of converting the sampled continuous analogue speech signal into a digital signal by expressing each sample as a finite number of digits. An error is introduced by representing the continuous signal by a finite number of binary values. This error is known as the quantization error or the quantization noise. In a digital computer the quantization levels are represented in powers of two (2^b), where b is known as the bits per sample of the ADC).

Coding is the process of assigning an unique binary number to each quantization level. Examples of coding techniques are the uniform quantizer and pulse code modulation

2.3.1 Sampling

The preprocessing stage is where the raw digital signal is further processed in order to improve the performance of the recognition system, or to prepare the speech for the feature calculation stage. A number of preprocessing methods follow:

2.3.2.1 Speech enhancement

In practical applications the environment in which the speech is recorded can be contaminated by background noise or interfering speakers. The transmission channel also deforms the speech.

The system implementer can either choose to ignore these effects at this stage and combat them on the feature level, or employ some kind of speech enhancement system.

In a practical system it would be to the advantage of the implementer to employ some kind of speech enhancement system.

2.3.2.2 Speech framing

In practical applications it is more convenient to work with short segments (frames) of the speech signal. Conventional analysis techniques typically require the signal to be at least wide sense stationary. The choice of the frame length should be made so that this assumption is reasonable.

Typical frame lengths vary between 15 ms and 40 ms in duration. The frames are usually overlapped by 50% to produce a smooth transition between frames.

2.3.2.3 Windowing

A window function, $w(n)$, is a real, finite length sequence (the length of the operating frame) that is multiplied by the signal (or frame).

A few commonly used windows such as Bartlett, Hanning, Hamming and Blackman are shown in Figure 2.2.

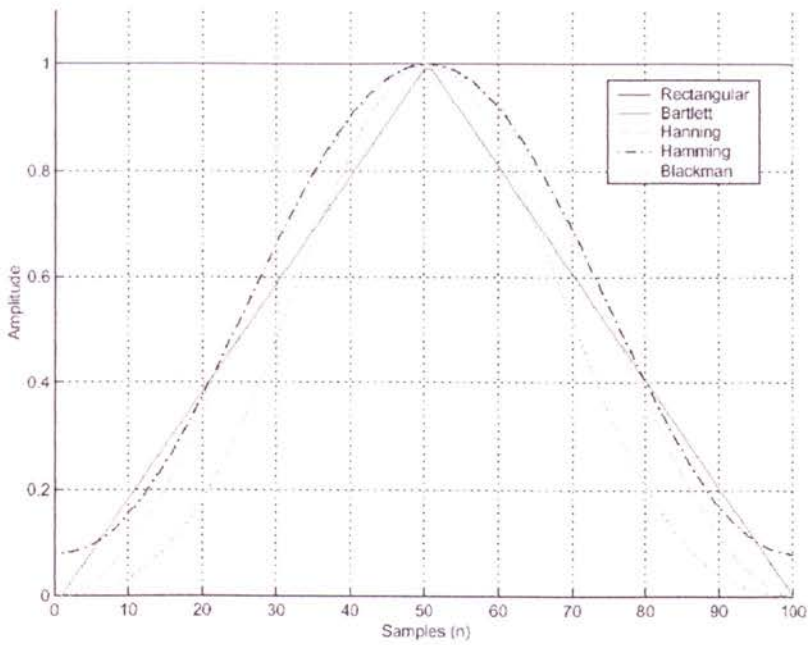


Figure 2.2: Time domain response of some common window

2.3.2.4 Pre-emphasis

A pre-emphasis filter is frequently applied to the speech signal before LP analysis.

It is typically a simple first order high pass filter that increases the relative energy of the higher frequency spectrum [7]. The transfer function of such a filter is

$$H(z) = 1 - \alpha z^{-1},$$

where α is a constant and $0.9 \leq \alpha \leq 1.0$.

There are several reasons for using a pre-emphasis filter. The first of these is that it is usually more desirable to only include the characteristics of the vocal tract for LP analysis. By applying a pre-emphasis filter the glottal waveform and lip radiation characteristics are eliminated.

The second reason is to prevent numerical instability. When calculating the high order LP coefficients, an ill conditioned autocorrelation matrix can result if the speech signal is dominated by low frequency components.

Finally, pre-emphasis also alleviates the effect of recording devices and the transmission of speech through air. Some recording devices attenuate the higher frequencies more than the lower ones and higher frequencies gets attenuated while propagating through air.

2.3.2.5 Endpoint Detection

It is crucial of finding the endpoints of speech in a waveform. If there was no silence removal, part of the voiceprint would come from the silence and an impostor could easily pass by just recording silence. So, it is imperative that the silent parts at the beginning and the end of the recording be deleted. The algorithm used to perform the endpoint detection is normally the zero-crossing rates.

The endpoint detection algorithm functions as follows:

1. The algorithm removes any DC offset in the signal. This is a very important step because the zero-crossing rate of the signal is calculated and plays a role in determining where unvoiced sections of speech exist. If the DC offset is not removed, the system will be unable to find the zero-crossing rate of noise in order to eliminate it from the signal.
2. Compute the average magnitude and zero-crossing rate of the signal as well as the average magnitude and zero-crossing rate of background noise. The average magnitude and zero-crossing rate of the noise is taken from the first hundred milliseconds of the signal. The means and standard deviations of both the average magnitude and zero-crossing rate of noise are calculated, enabling us to determine thresholds for each to separate the actual speech signal from the background noise.

3. At the beginning of the signal, search for the first point where the signal magnitude exceeds the previously set threshold for the average magnitude. This location marks the beginning of the voiced section of the speech.
4. From this point, search backwards until the magnitude drops below a lower magnitude threshold.
5. From here, search the previous twenty-five frames of the signal to locate if and when a point exists where the zero-crossing rate drops below the previously set threshold. This point, if it is found, demonstrates that the speech begins with an unvoiced sound and allows the algorithm to return a starting point for the speech, which includes any unvoiced section at the start of the phrase.
6. The above process will be repeated for the end of the speech signal to locate an endpoint for the speech.

2.3.3 Features Calculation

Speaker-specific characteristics can be classified into two level features, i.e., high-level features such as dialect, context, speaking style etc. and low level features such as pitch, spectral magnitudes, formant frequencies, etc. In speaker recognition system, high level features are seldom used because they are not easy to measure and not well understood. Low-level features are associated with physical and structural characteristics of the speaker and comparatively easier to quantify, therefore, most approaches for speaker recognition focus on the low-level features.

The effectiveness of the low level features has been studied extensively. These low-level features can be roughly grouped to spectral envelop-based features and prosodic features. Although many prosodic parameters such as pitch, formant frequencies etc. are effective features, they are not used alone for speaker recognition

because they are not robust due to the measurement difficulties. Currently, spectral envelope features are popular for speaker recognition system.

In prosodic parameters, voice pitch is a good feature to identify the speaker and has been employed in several speaker verification systems. However it is not easy to measure accurately, especially in noisy environments. In addition, it is easy to mimic and can be changed by other factors such as stress and speech effort levels. These defects make it less accurate to verify speaker when using this feature alone. Speech intensity has also been shown to be an effective feature. Formant frequencies, which are the resonant frequencies of the vocal tract, contain speaker-specific information and can be used to distinguish between speakers. The main drawback is the difficulty of measurement. Co-articulation during the production of a nasal sound has also been employed as a speaker specific feature.

Spectral envelop based features indicate the vocal tract characteristics, and therefore can be treated as efficient features for speaker recognition. This is because vocal tract is the main physiological aspect of the human speech production system. Besides, these envelope-based features can be easily drawn from the speech signal based on the linear prediction model. Spectral envelop based features include short-term and long-term spectrum features. The short-term spectrum of the speech signal, defined as a function of time, frequency, and spectral magnitude, is by far the most prevalent method of representation of the speech signal. Several approximations to the short-term spectrum, such as filter bank magnitudes, Linear Prediction Coding (LPC) spectral and cepstral coefficients, are also popular. The choice of a particular representation is determined by practical considerations, such as ease of computation, storage requirements, methods of pattern matching, susceptibility to transmission channel distortions, etc.

In the next section, detailed descriptions of mel-frequency cepstral coefficients (MFCC) is presented.

2.3.3.1 Mel Frequency Cepstral Coefficients (MFCC)

A block diagram of the structure of an MFCC processor is given in Figure 2.3. The speech input is typically recorded at a sampling rate above 10000 Hz. This sampling frequency was chosen to minimize the effects of *aliasing* in the analog-to-digital conversion. These sampled signals can capture all frequencies up to 5 kHz, which cover most energy of sounds that are generated by humans. As been discussed previously, the main purpose of the mfcc processor is to mimic the behavior of the human ears. In addition, rather than the speech waveforms themselves, mfcc's are shown to be less susceptible to mentioned variations.

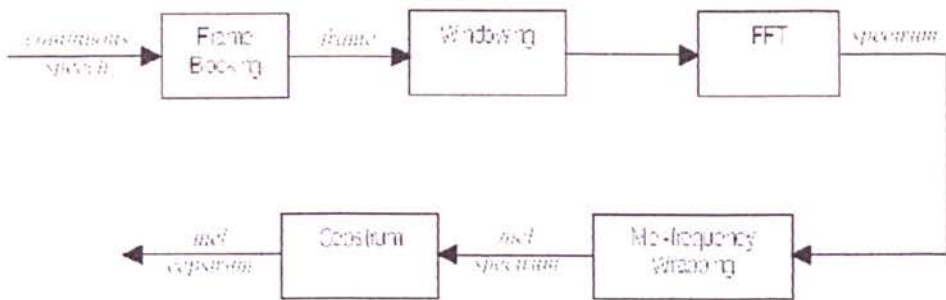


Figure 2.3: Block diagram of MFCC processor

Frame Blocking

In this step the continuous speech signal is blocked into frames of N samples, with adjacent frames being separated by M ($M < N$). The first frame consists of the first N samples. The second frame begins M samples after the first frame, and overlaps it by $N - M$ samples. Similarly, the third frame begins $2M$ samples after the first frame (or M samples after the second frame) and overlaps it by $N - 2M$ samples. This process continues until all the speech is accounted for within one or more frames. Typical values for N and M are $N = 256$ (which is

equivalent to ~30 msec windowing and facilitate the fast radix-2 FFT) and $M = 100$.

Windowing

The next step in the processing is to window each individual frame so as to minimize the signal discontinuities at the beginning and end of each frame. The concept here is to minimize the spectral distortion by using the window to taper the signal to zero at the beginning and end of each frame. If the is defined as $w(n)$, $0 \leq n \leq N - 1$, where N is the number of samples in each frame, then the result of windowing is the signal

$$y_l(n) = x_l(n)w(n), 0 \leq n \leq N - 1$$

Typically the *Hamming* window is used, which has the form:

$$w(n) = 0.54 - 0.46 \cos\left(\frac{2\pi n}{N-1}\right) \quad 0 \leq n \leq N-1$$

Fast Fourier Transform (FFT)

The next processing step is the Fast Fourier Transform, which converts each frame of N samples from the time domain into the frequency domain. The FFT is a fast algorithm to implement the Discrete Fourier Transform (DFT) which is defined on the set of N samples $\{x_n\}$, as follow:

$$X_n = \sum_{k=0}^{N-1} x_k e^{-j 2\pi k n / N}, \quad n = 0, 1, 2, \dots, N-1$$

Note that j is used here to denote the imaginary unit, i.e. $j = \sqrt{-1}$. In general X_n 's are complex numbers. The resulting sequence $\{X_n\}$ is interpreted as follow: the zero frequency corresponds to $n = 0$, positive frequencies $0 < f < F_s / 2$ correspond

to values $1 \leq n \leq N/2 - 1$, while negative frequencies $-F_s/2 < f < 0$ correspond to $N/2 + 1 \leq n \leq N - 1$. Here, F_s denotes the sampling frequency.

The result obtained after this step is often referred to as signal's *spectrum* or *periodogram*.

Mel-frequency Wrapping

As mentioned above, psychophysical studies have shown that human perception of the frequency contents of sounds for speech signals does not follow a linear scale. Thus for each tone with an actual frequency, f , measured in Hz, a subjective pitch is measured on a scale called the 'mel' scale. The *mel-frequency* scale is a linear frequency spacing below 1000 Hz and a logarithmic spacing above 1000 Hz. As a reference point, the pitch of a 1 kHz tone, 40 dB above the perceptual hearing threshold, is defined as 1000 mels. Therefore the following approximate formula to compute the mels for a given frequency f in Hz:

$$mel(f) = 2595 \log_{10} (1 + f/700)$$

One approach to simulating the subjective spectrum is to use a filter bank, one filter for each desired mel-frequency component (see Figure 2.4). That filter bank has a triangular bandpass frequency response, and the spacing as well as the bandwidth is determined by a constant mel-frequency interval. The modified spectrum of $S(w)$ thus consists of the output power of these filters when $S(w)$ is the input. The number of mel spectrum coefficients, K , is typically chosen as 12.

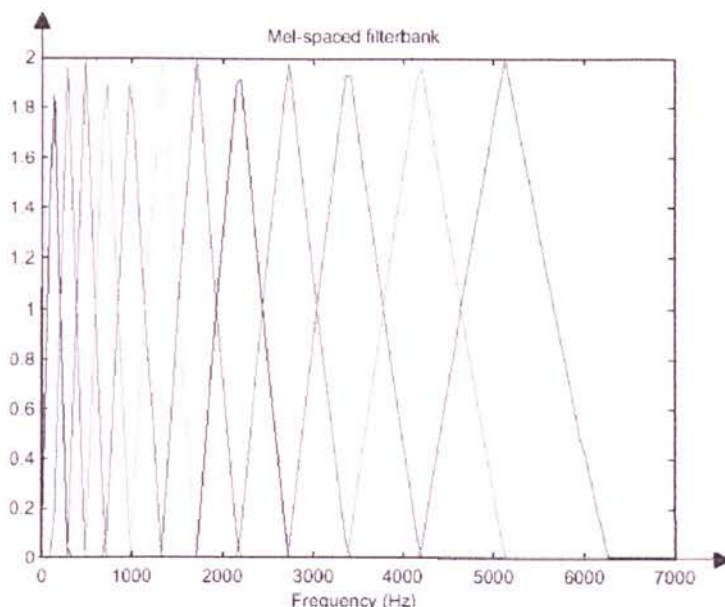


Figure 2.4. MFCC Filterbank

Note that this filter bank is applied in the frequency domain, therefore it simply amounts to taking those triangle-shape windows in the Figure 2.4 on the spectrum. A useful way of thinking about this mel-wrapping filter bank is to view each filter as an histogram bin (where bins have overlap) in the frequency domain.

Cepstrum

In this final step, the log mel spectrum is converted back to time. The result is called the mel frequency cepstrum coefficients (MFCC). The cepstral representation of the speech spectrum provides a good representation of the local spectral properties of the signal for the given frame analysis. Because the mel spectrum coefficients (and so their logarithm) are real numbers, using the Discrete Cosine Transform (DCT) can convert them to the time domain. Therefore if those mel power spectrum coefficients that are the result of the last step are denoted as S_k , $k = 1, 2, \dots, K$, calculate the MFCC's, \hat{C}_n as

$$\tilde{c}_n = \sum_{k=1}^K (\log \tilde{S}_k) \cos \left[n \left(k - \frac{1}{2} \right) \frac{\pi}{K} \right], \quad n = 1, 2, \dots, K$$

Note that the first component, \hat{C}_0 from the DCT are excluded since it represents the mean value of the input signal which carried little speaker specific information. By applying the procedure described above, for each speech frame of around 30msec with overlap, a set of mel-frequency cepstrum coefficients is computed.

2.3.4 Feature Analysis

2.3.4.1 Feature Normalization

The most significant factor affecting automatic speaker recognition performance is variation in the signal characteristics from trial to trial. Variations arise from the speaker themselves, from differences in recording and transmission conditions, and from background noise. Speakers cannot repeat an utterance precisely the same way from trial to trial. There are also long-term changes in voices. Therefore, it is important for speaker recognition systems to accommodate to these variations. Two types of normalization techniques: parameter domain normalization and the distance/similarity domain normalization are given as follows:

Parameter-Domain normalization

Spectral equalization is a typical parameter domain normalization technique. It is effective in reducing linear channel effects and long-term spectral variation. This method is especially effective for text-dependent speaker recognition applications that use sufficiently long utterances. With this normalization technique, cepstral coefficients are averaged over the duration of an entire utterance and the averaged values are subtracted from the cepstral coefficients of each frame. Additive variation in the log spectral domain can be compensated for fairly well by this method. However, it unavoidably removes some text-dependent and speaker specific features. Therefore, it is inappropriate for short utterances in speaker recognition applications.

Distance/Similarity-Domain normalization

Another normalization technique uses likelihood ratio to get the distance similarity. The likelihood ratio is defined as the ratio of two conditional probabilities of the observed measurements of the utterance, i.e., the ratio of the likelihood of the acoustic data given the claimed identity of the speaker and the likelihood given that the speaker is an imposter. The likelihood ratio normalization approximates optimal scoring in the Bayes sense.

A normalization method based on *a posteriori* probability has also been proposed. The difference between the normalization method based on the likelihood ratio and the method based on *a posteriori* probability is whether or not the claimed speaker is included in the speaker set for normalization; the speaker set used in the method based on the likelihood ratio does not include the claimed speaker, whereas the normalization term for the method based on *a posteriori* probability is calculated by using all the reference speakers, including the claimed speaker.

Experimental results indicate that the two normalization methods are almost equally effective. They both improve speaker separability and reduce the need for text-dependent or text-independent threshold.

2.3.4.2 Feature Comparison

There are several metrics that can be used for this comparison. Following will introduce Euclidean metric and a metric motivated by principle component analysis

Euclidean Metric

This metric is simply a Euclidean distance measurement. It gives a basis for comparing other methods.

The error for the Euclidean metric is defined as follows:

$$e = \sum_{v \in S} \|C(v) - v\|$$

where S is the set of feature vectors $\{v_1, v_2 \dots v_n\}$, and $C(v)$ is the cluster that is closest to v in Euclidean distance.

The PCA-Enhanced Metric

The PCA-Enhanced metric is a simple extension to the Euclidean metric. Principle component analysis is performed on all of the classified clusters of the test data, and creates a n -dimensional Gaussian probability function in space (n is the number of the clustering selected). Error function is then defined as:

$$e = \frac{1}{\|S\|} \sum_{v \in S} \prod_i ([C(v) - v]_i < k\sigma_i)$$

where i is the i th principle direction and the inequality is a “truth” value that is 1 if the expression is true and 0 if it is false. σ_i is the standard Gaussian deviation of $[C(v) - v]_i$ and k is a constant selected. Normally, k was selected as 2 because the probability of Gaussian distributed random variable greater than 2 standard deviations from the mean is very small. Here, assume $[C(v) - v]_i$ is Gaussian distributed.

2.4 Signal Classification

Classification has two distinct meanings. It may be given a set of observations with the aim of establishing the existence of classes or clusters in the data. Or there are for certain that there are so many classes, and the aim is to establish a rule whereby it can classify a new observation into one of the existing classes. The former type is known as Unsupervised Learning (or Clustering), the latter as Supervised Learning [1]

The task of *classification* occurs in a wide range of human activity. At its broadest, the term could cover any context in which some decision or forecast is made on the basis of currently available information, and a *classification procedure* is then some formal method for repeatedly making such judgments in new situations. Considering a more restricted interpretation, it is assumed that the problem concerns the construction of a procedure that will be applied to a continuing sequence of *cases*, in which each new case must be assigned to one of a set of pre-defined *classes* on the basis of observed *attributes* or *features*.

The construction of a classification procedure from a set of data for which the true classes are known has also been variously termed *pattern recognition*, *discrimination*, or *supervised learning* (in order to distinguish it from *unsupervised learning* or *clustering* in which the classes are inferred from the data).

2.4.1 Artificial Neural Network

2.4.1.1 Introduction

A neural net is an artificial representation of the human brain that tries to simulate its learning process. The term "artificial" means that neural nets are implemented in computer programs that are able to handle the large number of necessary calculations during the learning process.

2.4.1.2 Biological Neural Network and Artificial Neural Network

The human brain consists of a large number (more than a billion) of neural cells that process informations. Each cell works like a simple processor and only the massive interaction between all cells and their parallel processing makes the brain's abilities possible.

Below is a sketch of such a neural cell, called a neuron:

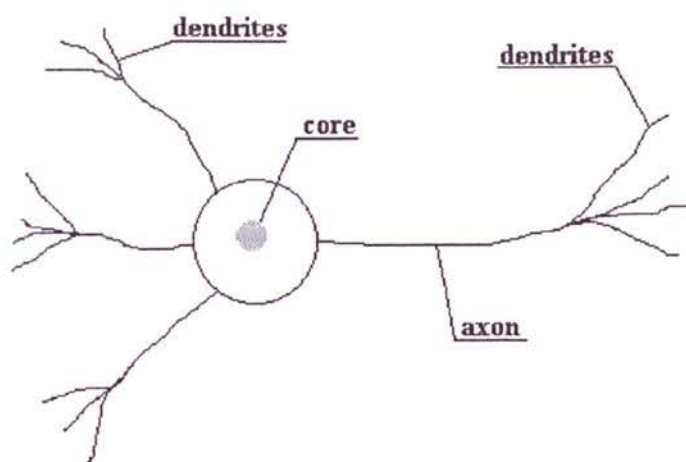


Figure 2.5: Structure of a neural cell in the human brain

Like the human brain, a neural net also consists of neurons and connections between them. The neurons are transporting incoming information on their outgoing connections to other neurons. In neural net terms these connections are called weights. The "electrical" information is simulated with specific values stored in those weights. By simply changing these weight values the changing of the connection structure can also be simulated.

The following figure shows an idealized neuron of a neural net.

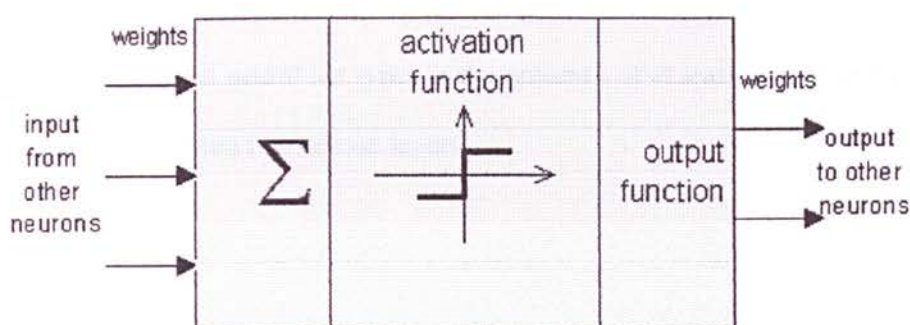


Figure 2.6: Structure of a neuron in a neural net

As can be seen, an artificial neuron looks similar to a biological neural cell. And it works in the same way. Information (called the input) is sent to the neuron on its

incoming weights. This input is processed by a propagation function that adds up the values of all incoming weights.

The resulting value is compared with a certain threshold value by the neuron's activation function. If the input exceeds the threshold value, the neuron will be activated, otherwise it will be inhibited. If activated, the neuron sends an output on its outgoing weights to all connected neurons and so on.

In a neural net, the neurons are grouped in layers, called neuron layers. Usually each neuron of one layer is connected to all neurons of the preceding and the following layer (except the input layer and the output layer of the net). The information given to a neural net is propagated layer-by-layer from input layer to output layer through either none, one or more hidden layers. Depending on the learning algorithm, it is also possible that information is propagated backwards through the net.

Figure 2.7 shows a neural net with three neuron layers.

Note that this is not the general structure of a neural net. For example, some neural net types have no hidden layers or the neurons in a layer are arranged as a matrix.

What's common to all neural net types is the presence of at least one weight matrix, the connections between two neuron layers

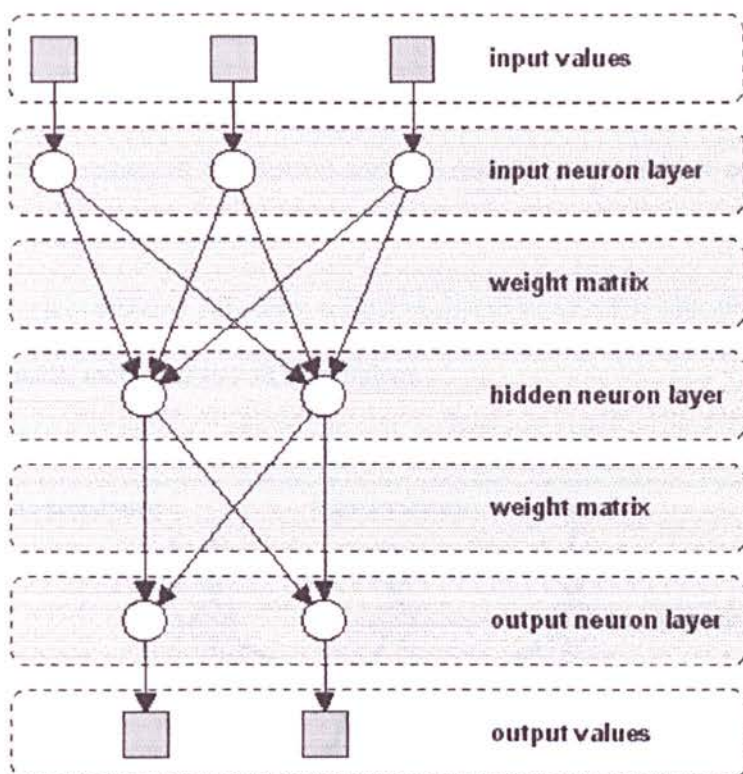


Figure 2.7: Neural net with three neuron layers

2.4.1.3 Neural Network Types

As mentioned before, several types of neural nets exist. They can be distinguished by their type (feedforward or feedback), their structure and the learning algorithm they use.

The *type* of a neural net indicates, if the neurons of one of the net's layers may be connected among each other. Feedforward neural nets allow only neuron connections between two different layers, while nets of the feedback type have also connections between neurons of the same layer.

In this section, only the Multi Layer Perceptron and its variant, the backpropagation Network is described since the task of classifying speaker voices had been done using these networks.

Multi-Layer-Perceptron

The Multi-Layer-Perceptron was first introduced by M. Minsky and S. Papert in 1969. It is an extended Perceptron and has one or more hidden neuron layers between its input and output layers.

Due to its extended structure, a Multi-Layer-Perceptron is able to solve every logical operation, including the XOR problem.

Multi-Layer-Perceptron characteristics	
sample structure	<p>The diagram illustrates a feedforward neural network with three layers: an input layer with 3 nodes, a hidden layer with 2 nodes, and an output layer with 3 nodes. Arrows show the flow of information from input values through weight matrix 1 to the hidden layer, then through weight matrix 2 to the output layer, resulting in output values.</p>
type	feedforward
neuron layers	1 input layer 1 or more hidden layers 1 output layer
input value types	binary
activation function	hard limiter / sigmoid
learning method	supervised
learning algorithm	delta learning rule backpropagation (mostly used)
mainly used in	complex logical operations pattern classification

Table 2.2: Multi-Layer-Perceptron

Backpropagation Net

The Backpropagation Net was first introduced by G.E. Hinton, E. Rumelhart and R.J. Williams in 1986 and is one of the most powerful neural net types. It has the same structure as the Multi-Layer-Perceptron and uses the backpropagation learning algorithm.

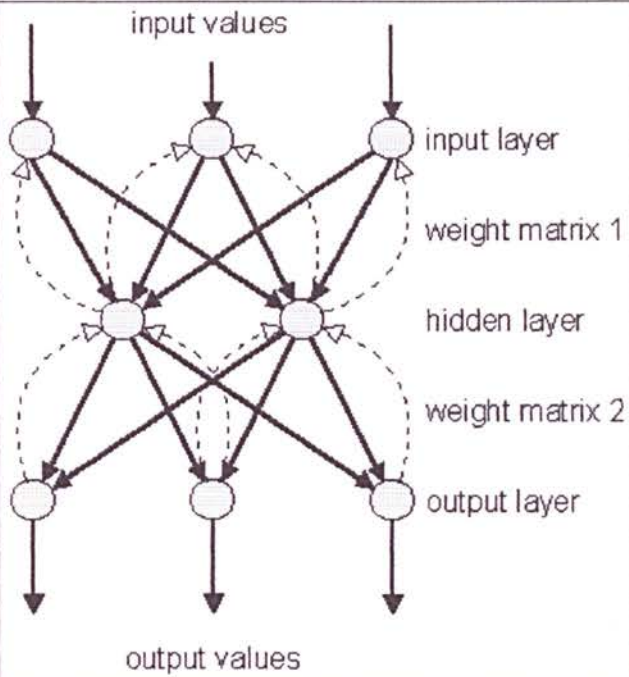
Backpropagation Net characteristics	
sample structure	 <p>The diagram illustrates a feedforward neural network with three layers: an input layer with 3 nodes, a hidden layer with 2 nodes, and an output layer with 3 nodes. Arrows labeled 'input values' point into the input layer nodes. Solid arrows represent forward connections from the input layer to the hidden layer (labeled 'weight matrix 1') and from the hidden layer to the output layer (labeled 'weight matrix 2'). Dashed arrows represent backward connections for error propagation from the output layer back to the hidden layer, and from the hidden layer back to the input layer. Arrows labeled 'output values' point out from the output layer nodes.</p>
type	feedforward
neuron layers	1 input layer 1 or more hidden layers 1 output layer
input value types	binary
activation function	sigmoid
learning method	supervised
learning algorithm	backpropagation
mainly used in	complex logical operations pattern classification speech analysis

Table 2.3: Backpropagation Net

2.4.2 Statistical/Stochastic Methods

The stochastic model treats the speech production process as a parametric random process and assumes that the parameters of the underlying stochastic process can be estimated in a precise, well defined manner. Recent work in stochastic models has demonstrated that these models are more flexible and hence allow for better modeling of the speech production process than template models.

2.4.2.1 Hidden Markov Models

Hidden Markov Models (HMM) are extensions to the conventional Markov models, wherein the observations are a probabilistic function of the state, i.e., the model is a doubly embedded stochastic process where the underlying stochastic process is not directly observable (it is hidden). The HMM can only be viewed through another set of stochastic processes that produce the sequence of observations. Thus, the HMM is a finite-state machine, where a probability density function $p(x | s_i)$ is associated with each state s_i . The states are connected by a transition network, where the state transition probabilities are $a_{ij} = p(s_i | s_j)$. A fully connected three-state HMM is depicted in Figure 2.8

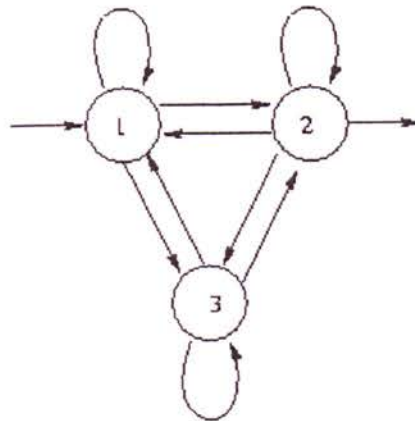


Figure 2.8: A fully connected three state HMM

For speech signals, another type of HMM, called a left-right model or a Bakis model, is found to be more useful. A left-right model has the property that as time increases, the state index increases or stays the same that is the system states proceed from left to right. Since the properties of a speech signal change over time in a successive manner, this model is very well suited for modeling the speech production process.

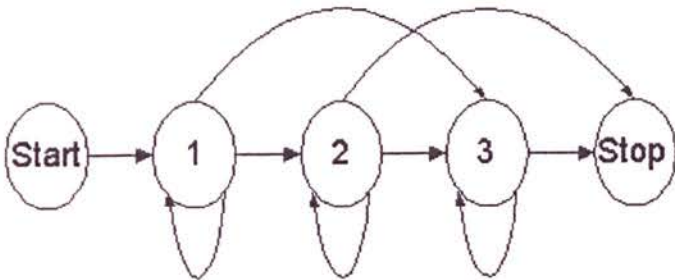


Figure 2.9: A left-to-right HMM model

For the Hidden Markov models discussed above, the matching score is the probability that a given set of feature vectors was generated by the model.

2.4.3. Template Methods

2.4.3.1 Vector Quantizations

Vector quantization (VQ) is basically a method of compressing the training data to a manageable and efficient size. VQ maps vectors from a large vector space to a finite number of regions in that space. Each region is called a *cluster* and can be represented by its center called a *codeword*. The collection of all codewords is called a *codebook*.

Figure 2.10 shows a conceptual diagram to illustrate this recognition process. In the figure, only two speakers and two dimensions of the acoustic space are shown. The circles refer to the acoustic vectors from the speaker 1 while the triangles are from the speaker 2. In the training phase, a speaker-specific VQ codebook is

generated for each known speaker by clustering his/her training acoustic vectors. The result codewords(centroids) are shown in Figure 2.10 by black circles and black triangles for speaker 1 and 2, respectively. The distance from a vector to the closest codeword of a codebook is called a VQ-distortion. In the recognition phase, an input utterance of an unknown voice is “vector-quantized” using each trained codebook and the *total VQ distortion* is computed. The speaker corresponding to the VQ codebook with smallest total distortion is identified.

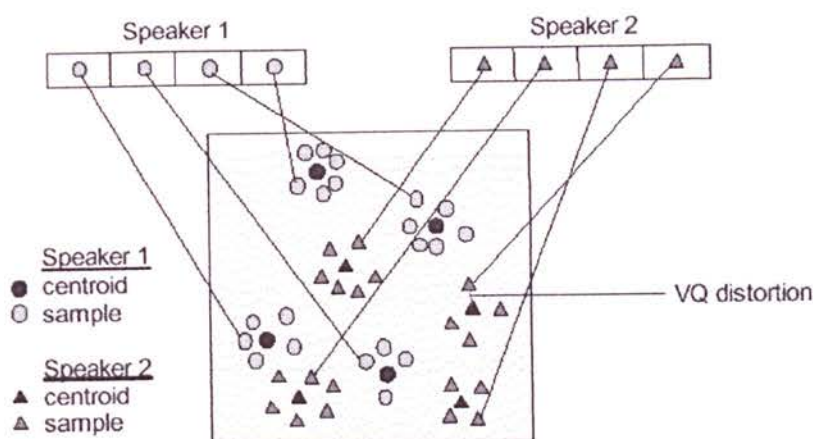


Figure 2.10. Conceptual diagram illustrating vector quantization codebook formation. One speaker can be discriminated from another based of the location of centroids.

Clustering the Training Vectors

There is a well-known algorithm, namely LBG algorithm [Linde, Buzo and Gray, 1980], for clustering a set of L training vectors into a set of M codebook vectors. The algorithm is formally implemented by the following recursive procedure:

1. Design a 1-vector codebook; this is the centroid of the entire set of training vectors (hence, no iteration is required here).
2. Double the size of the codebook by splitting each current codebook y_n according to the rule

$$y_n^+ = y_n(1 + \varepsilon)$$

$$y_n^- = y_n(1 - \varepsilon)$$

where n varies from 1 to the current size of the codebook, and ε is a splitting parameter ($\varepsilon = 0.01$ is chosen).

3. Nearest-Neighbor Search: for each training vector, find the codeword in the current codebook that is closest (in terms of similarity measurement), and assign that vector to the corresponding cell (associated with the closest codeword).
4. Centroid Update: update the codeword in each cell using the centroid of the training vectors assigned to that cell.
5. Iteration 1: repeat steps 3 and 4 until the average distance falls below a preset threshold.
6. Iteration 2: repeat steps 2, 3 and 4 until a codebook size of M is designed.

Intuitively, the LBG algorithm designs an M -vector codebook in stages. It starts first by designing a 1-vector codebook, then uses a splitting technique on the codewords to initialize the search for a 2-vector codebook, and continues the splitting process until the desired M -vector codebook is obtained.

Figure 2.11 shows, in a flow diagram, the detailed steps of the LBG algorithm. “*Cluster vectors*” is the nearest-neighbor search procedure which assigns each training vector to a cluster associated with the closest codeword. “*Find centroids*” is the centroid update procedure. “*Compute D (distortion)*” sums the distances of all training vectors in the nearest-neighbor search so as to determine whether the procedure has converged.

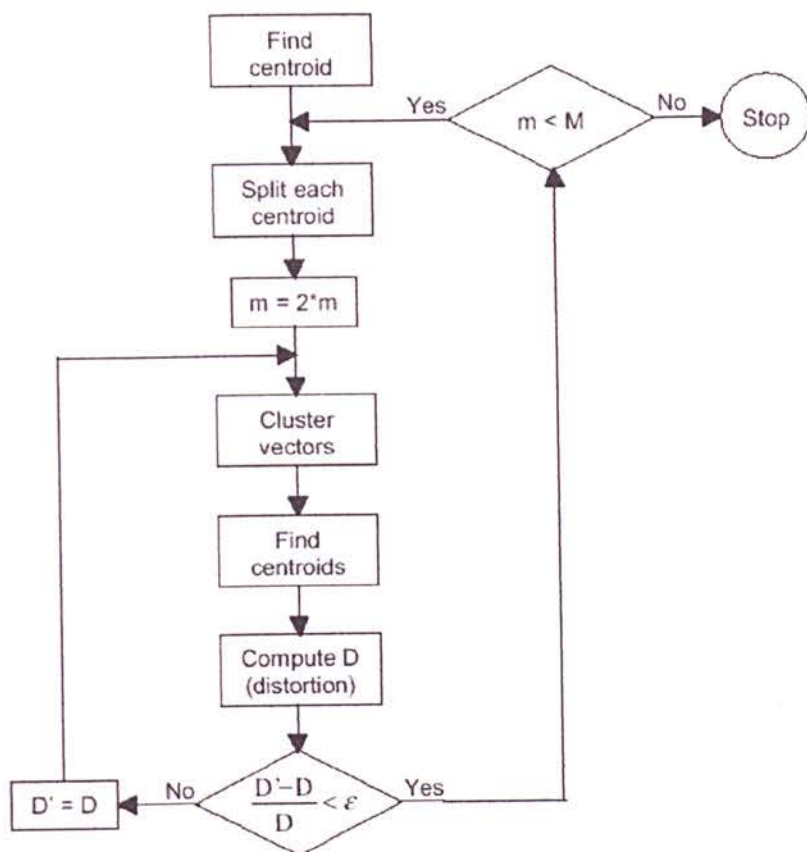


Figure 2.11 Flow diagram of the LBG algorithm

The model's linkage to the left side with the right side of the V implies that if problems are found during verification and validation, then the left side of the V can be re-executed to fix and improve the requirements, design, and code before the testing steps on the right are reenacted.

3.2.1 Requirement Analysis

The main goal for this system analysis is to really understand the proposed system and establishing the system requirement.

Sources from published materials, the internet and from examination of other available speaker verification system were gathered and analyzed to model the information and functional domain at the same time. The outcome will be an accurate system requirement specification.

3.2.2 System & Program Design

This phase will translate the system requirement that was produced from the previous phase into a representation of a system.

3.2.3 Coding

This is the phase where the design representation will be realized into a working system. The coding phase is critical where all the considerations taken into account will yield results and where mistakes will be costly .

3.2.4 Testing

This is a critical phase as it assures the quality of the developed system. Any mistakes in the testing means that the phase at the left side of the V-Model is faulty and should be redone.

During unit and integration testing, all aspects of the program design are ensured to be implemented correctly in the code. System testing should verify the

system design, making sure that all system design aspects are correctly implemented. Acceptance testing is done by the customer to check that all requirements have been fully implemented.

3.2.5 Operation and Maintenance

In this phase, the system will be maintained and its operation monitored.

CHAPTER 4

SYSTEM ANALYSIS

4.1 Functional Requirements

Functional requirements are, simply say, requirements that must be met in the development of the system and required for the system to function correctly. The next few section will cover the functional requirements for the proposed system.

4.1.1 Voice Acquisition

The system must be able to provide a simple procedure to acquire the voices of the user. In the enrollment module, the user will be asked to pronounce his passphrase two times and for the verification module, just once.

Users are allowed to choose his own passphrase, regardless of language and meaning.

4.1.2 Speech Signal Pre-processing

The proposed system must be able to adjust an input speech signal to be in a better quality and to have appropriate characteristics for the next processing step. Analogue speech signal needs to be converted to a digital form. Endpoint detection is also done on the waveform so that silence in the beginning and the end of the speech signal is deleted.

It is this functionality that will determine whether the system is robust enough against background noise and channel noise.

4.1.3 Feature Extraction

The heart of any speaker verification system is to extract speaker dependent features from the speech which should ideally have following characteristics.

- Easily measurable
- Vary much as possible among the speakers, but be consistent within each speaker.
- Not change over time or be affected by the speaker's health.
- Not be affected by background noise nor depend on the specific transmission medium.
- Not be modifiable by conscious effort of the speaker or at least, be unlikely to be affected by attempts to disguise the voice.
- Occur naturally and frequently in speech.

4.1.4 Features Clustering

The system will use a clustering algorithm to differentiate the features between the speakers because feature vectors from a given speaker tend to cluster closely amongst them.

In addition, the system determines the decision threshold score to be used during the verification process.

4.1.5 Verification

The system must produce a result from the verification process which is either accept or reject.

4.2 Non-functional Requirements

Non-functional requirements is an essential definition of a system properties and constraints under which a system must operate. Although the non-functional requirements are subjective, they are as important as the functional requirements.

They are a few aspects to be considered such as;

4.2.1 System Reliability

The system must be reliable enough to allow the right user to access the system while impostors are denied access.

4.2.2 User-friendly Interface

The user interface of the application must be simple but attractive and must be suited in its use.

4.2.3 Response Time

An application involving authentication is supposed to have a fast response time or provide results at an acceptable interval in order not to cause the user to wait for a long time to have access or make their transaction

4.2.4 Robustness

The voice authentication must be robust against noise emitted in the background and provide the right result even when there is noise.

4.3 Technical Requirements – Hardware

In developing the proposed system, the configuration of the hardware to be used in the development process is as follows:

- Standard PC
- Microphone

4.4 Technical Requirements – Software

4.4.1 Development Tools

These are the proposed development software to be used to develop the speaker verification system :

1. Matlab 6.1 :-

- Matlab will be used to build both of the classifier since Matlab can be used to handle signal processing and classification of signals.

4.4.2 Operating System

The operating system that will be used as a platform to build the application is

Microsoft Windows Me.

CHAPTER 5

SYSTEM DESIGN & ARCHITECTURE

5.1 System Design

5.1.1 Voice Acquisition

This is the process where voices are obtained. Enrollments are done by the user by pronouncing or saying his or her password 2 times. The first is to create the codebook and the second is to create the user's threshold.

5.1.2 Speech Signal Pre-Processing & Feature Extraction

In the preprocessing, the zero crossing algorithm will be used to perform the silence detection and cutting out the speech signal from the original signal. For the feature extraction process, the Mel-Frequency Cepstral Coefficients (MFCC) [refer Section 2.3.3.3] will be used as the feature detection.

5.1.3 Clustering and Pattern Matching

The clustering is the most important aspect of this system since this is where the calculations are done to distinguish between the voices of the users.

The vector quantization approach that was discussed in 2.3.4.1 will be used for the development of this module. The algorithm discussed which is the Linde-Buzo-Gray (LBG) will be used.

5.1.4 Threshold Creation

Since some words can be said more consistently than others and people say things with different consistency, it makes sense to have user specific thresholds. After users have created their voiceprint, they are asked to make a second recording.

The system takes this second recording and calculates the difference, or average distortion, between the second recording and the saved voiceprint. Average distortion is the average euclidean distance between the test vectors and the codebook vectors. This distortion is then used as a basis for the the threshold. Specifically, the distortion from the second recording is multiplied by 1.2 and saved as the threshold.

The reasoning behind the multiplication factor is that the user is likely to say the word more differently the third time around during verification.

5.1.5 Verification

Verification is where the real test of the system to differentiate the user is. A comparison is then made between the just recorded voice and the voiceprint of the user. If the average distortion is higher than the threshold, then the user fails the verification; if it is lower, then they pass. The comparison, like in the threshold creation module is done by calculating the average euclidean distance between the test vectors and codeword vector(centroid) and then comparing it with the threshold.

5.2 System Flow Diagram

5.2.1 Enrollment Module

This is the data flow chart for the enrollment module.

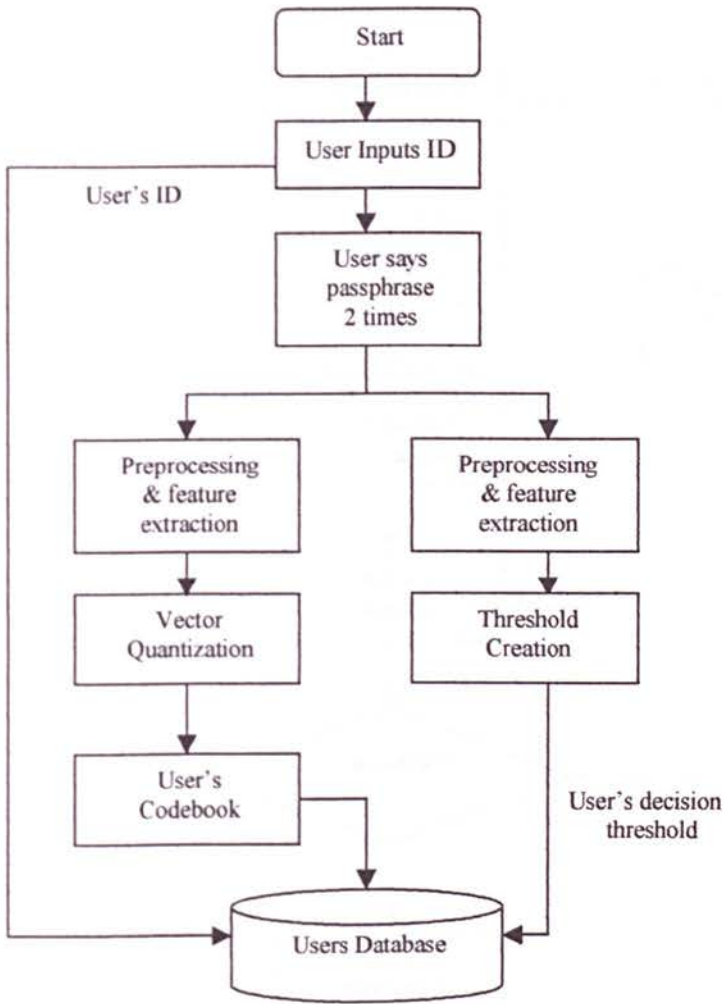


Figure 5.1: Flow chart of enrollment module

5.2.2 Verification Module

This is the data flow chart for the verification module

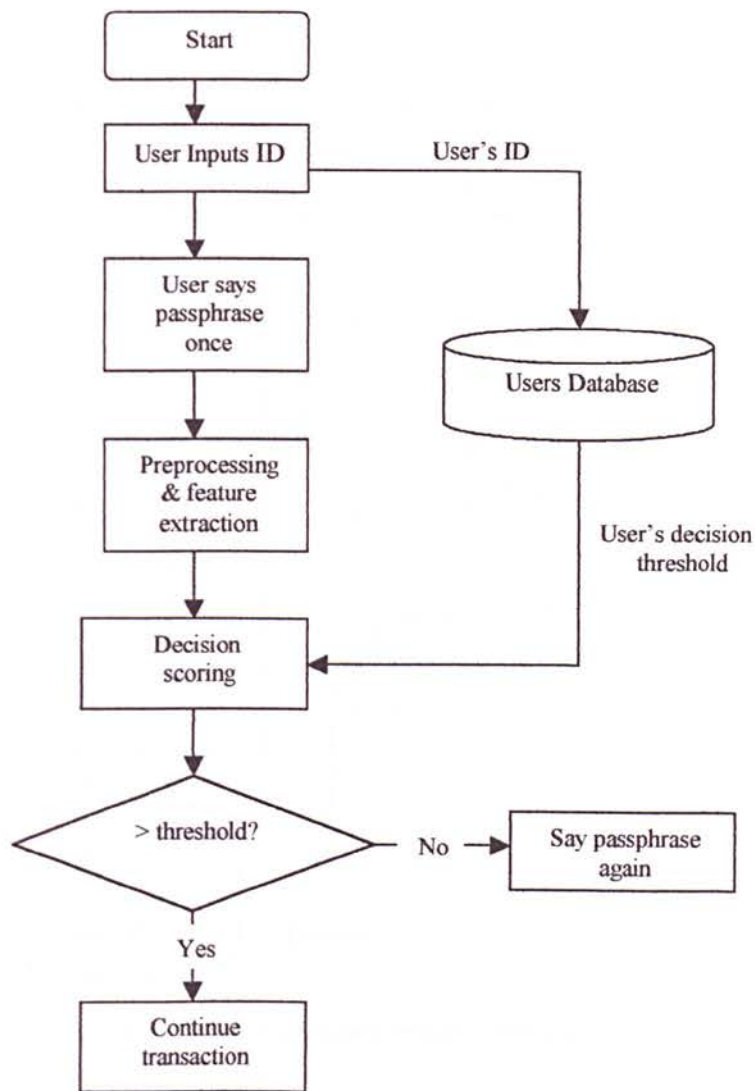


Figure 5.2: Flow chart of verification module

5.3 System Architecture

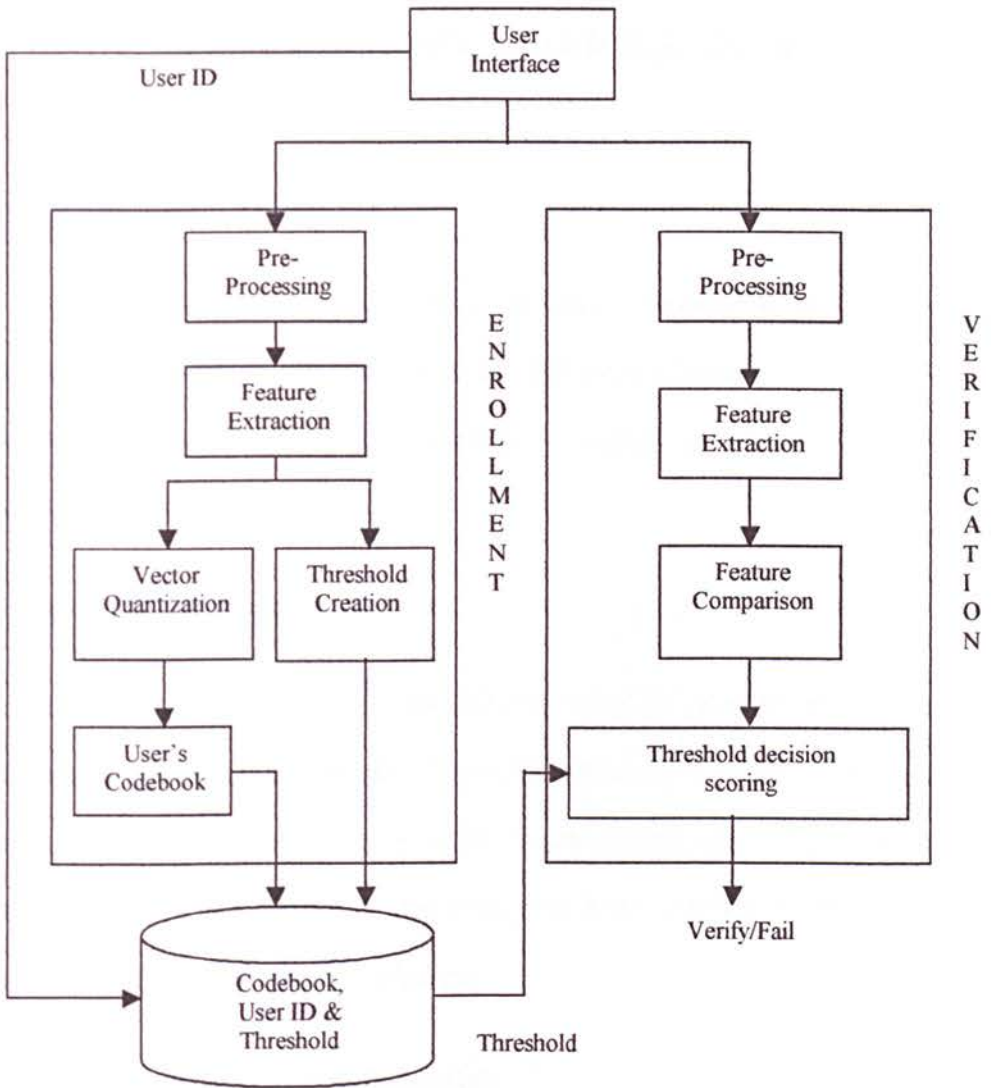


Figure 5.3: The proposed Speaker Verification system architecture flow

CHAPTER 6

SYSTEM IMPLEMENTATION

6.1 Platform

Matlab was chosen as the platform for ease of implementation. The system functionality is available using Matlab 5 but Matlab 6 is required for the GUI. The Matlab Data Acquisition Toolbox (DAQ) is needed to perform the program recording.

6.2 Program Usage

The section covers the function calls for using the program from the Matlab command line. Please see Appendix B for GUI screenshots. The main function is to access the modules of the system which is enrollment, threshold creation, and verification. The calling syntax is mainmenu. Available options from the main menu are User Enrollment and User Verification.

6.3 High Level Organization

There is nearly a one to one correspondence between the design modules and implemented functions. Basic functionality from the design was implemented into the core functions accessible through the Matlab command-line. See Appendix C for files created and the calling structure.

6.4 Functional Details

6.4.1 Signal Preprocessing

6.4.1.1 Endpoint Detection

The implementation for the speaker verification system first addresses the issue of finding the endpoints of speech in a waveform. The code which executes the algorithm can be found in the file *locatespeech.m*. The algorithm removes any DC offset in the signal. If the DC offset is not removed, the zero-crossing rate of noise cannot be found in order to eliminate it from the signal. The algorithm finds the start and end of speech in a given waveform, allowing the speech to be removed and analyzed.

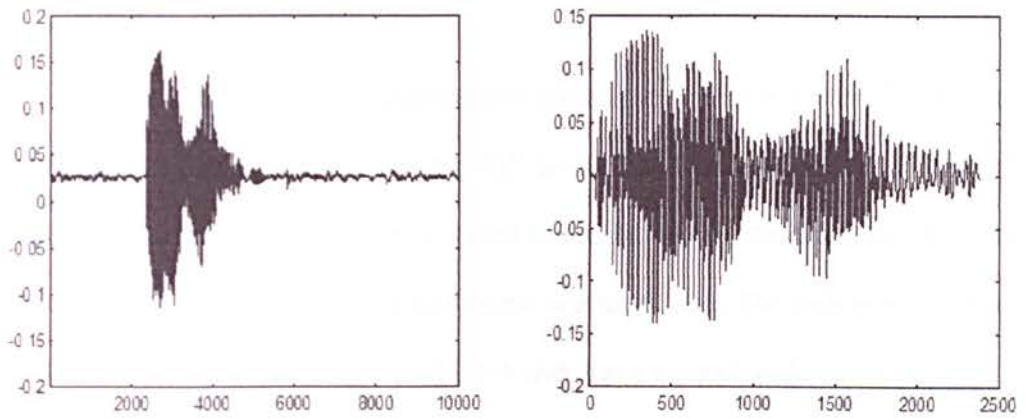


Fig. 6.1. Example of speech period extraction

6.4.2 MFCC

The Enrollment module was used to call the functions to create the MFCC's. The function implementation of extracting the MFCC's uses a filterbank to extract the power coefficients at specific mel-scale points is *melcepst.m*. Using the filterbank essentially mimics the way the human ear perceives certain frequencies.

The function requires the following parameters: signal, sampling frequency, window type, number of coefficients, number of filters in the filterbank, length of a frame, and the frame increment. Default values are shown in Table 6.1.

Parameter	Default Value
Sampling Frequency	16kHz
Window Type	Hamming
Number of Coefficients (in each frame)	12
Filters in filterbank	29
Length of the frame	256
Frame increment	128

Table 6.1. Default Mel-Cepstrum Parameters

Note that the mel-cepstrum function uses a real FFT with a frequency resolution equal to the length of the frame.

6.4.3 Vector Quantization

The functions for vector quantization are *vqlbg.m* and *kmeans.m*. Given a set of MFCC's, and the number of desired codeword, *vqlbg.m* outputs a list of codeword. The function *kmeans.m* is called by *vqlbg.m* to perform the iterations of K-nearest neighbor for the amount of desired codebook size. The codeword are then saved into a .mat file for later retrieval. Each user has a separate codeword file.

6.4.4 Threshold Creation

A global and static threshold could not be defined. A threshold was needed for each user. In order to fix this, a third pseudo-module was implemented for threshold generation. The function for threshold generation is *id_threshold.m*. Even though this appears to be a separate module, from the user GUI perspective, it is part of the User Enrollment module. What this new module does is it performs a User Verification session right after the User Enrollment session and then calculate the threshold. This is done through the "record 2" button on the GUI. Refer to Appendix A.

6.4.5 Decision

The verification decision is made by calculating the average Euclidean distance between the test vectors and codeword. This is done in the Verification module, performed by the function *id_test_verify.m* which calls the Euclidean distance calculation function. The average Euclidean distance is then compared to the user's threshold for the pass/fail verdict.

CHAPTER 7

SYSTEM TESTING

The test phase was broken into three smaller phases: Integration Test, System Test, and Performance Test.

7.1 Integration Test

The goal of this test phase was to ensure that each subsystem of the overall system worked together. This phase was necessary because each subsystem in the overall system was assigned an individual in the group for implementation. During this phase, test scripts are generated to see if every adjacent subsystem communicated properly. Due to the extra effort spent on the System Design phase, the input/output specifications in each module were clear enough to allow easy integration.

7.2 System Test

The goal of this test phase was to ensure that the overall system performed to the functionality specifications.

The goal was to verify that a User Enrollment session, and a User Verification session could be held. Both sessions ran properly on the first test. Codebook was generated successfully during the User Enrollment session. The threshold was created successfully and the results are stored properly. An average distortion factor was generated properly in the User Verification session. The verdict subsystem was disabled for the time being and the average distortion factor was observed directly.

This was done because the value of the average distortion factor was uncertain at this point.

7.2.1 Problems encountered during system testing

The main problem that was found during the system test is when the system sometimes stops after a user has spoken into the microphone. An error message was displayed in the Matlab command line, which is '*Matrix dimension does not match*'. This, however, is not a fatal error and the user can still click the record button again to re-record their voices until the correct message is displayed.

The reason for this problem is probably due to faulty microphone or when the sound recorded exceeds a certain maximum decibel value and the signal is too convoluted. An example is when the user breathes into the microphone while recording his password. Currently the problem is not yet solved.

7.3 Performance Test

The goal of this test phase was to find optimal values for certain parameters in the system, the accuracy of the system, speed of execution and disk space usage.

7.3.1 Optimization Tests

The parameters under test included codebook size, number of MFCC's per acoustic vector, and threshold generation scaling factor. Test scripts were written to vary each parameter and observe the average distortion generated each time.

Optimality for the threshold generation scaling factor was determined to minimize false acceptances and false rejections. Tests showed that if the scaling factor was too high, there were significantly more false acceptances. And if the scaling factor was too low, there were significantly more false rejections. This is obvious since the threshold is the sensitivity level. After careful testing, a scaling

factor of 1.2 was determined to be the optimal value. Optimality for the other two was determined based on a speed vs. accuracy trade-off. The optimal values are summarized in Table 7.1.

Parameter	Optimal Value
No. of MFCC's per acoustic vector	12
No. of Codewords	64
Threshold Generation Scaling Factor	1.2

Table 7.1. Optimal Parameter Values

7.3.2 Execution Time Test

The execution speed of the User Enrollment, Threshold Generation, and User Verification modules were analyzed. It was found that the time needed to perform the codebook creation in the Enrollment module took the most time compared to the others while the threshold creation took the least time.

7.3.3 Disk Space Usage

The codebook size for a 64-codeword codebook and 12 MFCC's required about 7 kilobytes of disk space. The threshold value for each user required 9 bytes.

Three .wav files are created and reused for the system which is User1st.wav, User2nd.wav and UserVerif.wav. Each wave file uses up about 63 kilobytes of space.

CHAPTER 8

SYSTEM DISCUSSION & RECOMMENDATIONS

In this chapter, a few topics of discussion will be brought up. The main discussion will be the performance of the system with all its strength and weaknesses. Other discussions are alternative options in developing the system and future measures recommendations that could be developed.

8.1 System Discussions

8.1.1 Enrollment Module

This system was developed for the purpose of using a chosen method to be used in recognizing a user's voice. This module does just that and quite successfully because there is no problem in creating the codebook.

The codebook that was created is stored into a file with the extension .mat and the filename is the UserID that the speaker entered. Note that the wave file (User1st.wav) that was created is used globally. Every time a new speaker enrolls, their voice will be recorded into the same wave file so that the old one will be replaced. The only trace of the previous speaker's wave file have been turned into a codebook in the UserID.mat. This was done to reduce the amount of storage space this system will use.

8.1.2 Threshold Module

The threshold module creates the threshold to be compared with to verify a given speaker. This threshold is generated automatically by multiplying the average distortion (the average euclidean distance between the test vectors and the codebook

vectors) by 1.2. This is a fixed scaling value. However, maybe in future implementations this scaling value can be chosen by the speaker as options for High, Normal or Low threshold.

It is important during the threshold creation to say the word with fair accuracy because if the resulting threshold is very high, imposters will have an easier time passing verification test.

In the threshold module, a new variable is inserted into the existing UserID.mat by the name thresh and the file is updated and saved. This is the threshold value to be used to decide the pass/fail verdict.

8.1.3 Verification Module

The verification module is where the pass/fail verdict is given by comparing the average distortion values from the feature vectors to the threshold value. If the verdict is a rejected verification, then just repeat until the verification is successful.

From the results obtained in verification testing, it was found that the False Rejection Rate (FRR) where the correct speaker is rejected is quite high. The False Acceptance Rate (FAR) where an impostor is verified successfully, however, is very low. From sources of research that was referenced, the cause of the high rejection rate is probably due to the length of the user's spoken password. A longer utterance will normally yield less FRR.

8.1.4 Microphone Issues

A very sensitive microphone will quite often results in the problem that was discussed in section 7.2.1. When this error happens, try to change the microphone or just click the record button again to re-record to get to the desired results. Try to avoid breathing into the microphone because this will convolute the speech signal. Currently, there is no other way around this problem.

8.2 Alternative Options

There is more than one way to perform speaker verification. The methods chosen for this project were mostly chosen because of their implementability and low complexity. Areas where the major options exist are in the feature extraction and verification modules. The list of alternatives below is in no way a complete listing.

Please see the references to find out more.

8.2.1 Feature Extraction Alternatives

Linear Prediction Coefficients and its derivatives.

Cepstrum: Identifies the vocal track parameters. Used for text-independent recognition.

Delta-Cepstrum: Analyses changing tones

8.2.2 Verification Alternatives

Dynamic Time Warping: Accounts for inconsistencies in the rate of speech by stretching or compressing parts of the signal in the time domain

Stochastic methods: Hidden Markov Models, Gaussian Mixture Models

AI based: Neural Networks

8.3 Recommendations

8.3.1 Threshold Generation Using Multiple Verification Sessions

Currently, the threshold is set based on the average distortion calculated by one verification session. If for some reason there is a large variance in that one session, then in future verification sessions, there may be very high false acceptances or very high false rejections.

To account for this, several verification sessions should be held and the average distortion factors from each session should be summed and averaged before scaling as the threshold.

8.3.2 Weighting the Code Vectors

If the codebooks for several users were compared, it can be seen that certain code words are generated relatively close to each other among the different users. Accuracy could be improved by creating some sort of weighting for each code word in each codebook. This way, code words that occur frequently would be weight less in the average distortion calculations.

8.3.3 Code Book Adaptation Over Time

An idea that was considered was allowing the codebook of each user to adapt over successful verification sessions. This would modify the thresholds and codebooks each time the user successfully passed the verification test. However, it was debated on whether it would then be plausible to convert a given user's codebook to match that of a different user. Thus defeating the purpose of the system.

8.3.4 Signal Normalization

In the Mel-Frequency transform, the output of the filterbanks depends on the power of the signal. This implies that speaking loudly will be seen differently than quietly. By normalizing the recording signal, this effect can be reduced.

8.3.5 Verification Threshold Options

As stated in the verification module discussion, the FRR rate is quite high for this system so another way to successfully verify a given speaker is by generating a low, normal and high threshold by multiplying the average distortion with other scaling values.

CONCLUSION

Overall, the project can be considered a success with the basic requirements being satisfied. The finished product could enroll users, verify their voiceprint, and provided a gui interface for users to do so. Matlab 6 is required to run the full program.

Performance of the system was fair with a false rejection rate of 23% and a false acceptance rate of 9.6%. False rejection errors are the result of the system not being able to overlook the small changes in a person's voice recording, for example, the emphasis that the put onto syllables or the changing tone of their voice. False acceptance errors occur when the imposter's voice has similar frequency characteristics to the true user. Adding threshold generation using multiple recordings and weighting of the code vectors would improve both false acceptance and rejection ratios while codebook adaptation and signal normalization would improve false rejection ratios.

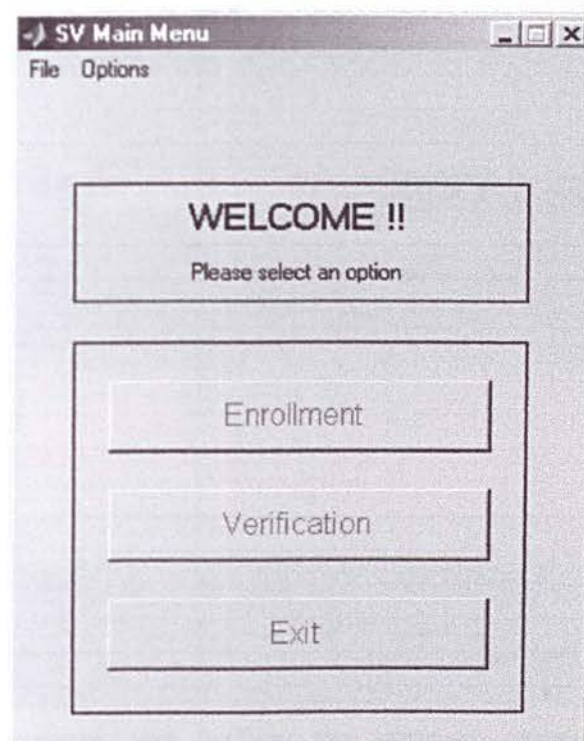
LIST OF REFERENCES

1. D. Michie(ed.), D.J. Spiegelhalter(ed.), C.C. Taylor(ed.), (1994). Machine Learning, Neural and Statistical Classification, e-book.
2. Shari Lawrence Pfleeger (2001), Software Engineering – Theory and Practice, United States of America: Prentice Hall, International, Inc.
3. R. O. Duda, P. E. Hart, D. G. Stork , (2001), Pattern Classification (2nd ed), John Wiley & Sons, Inc.
4. B. Gold, N. Morgan (2000); Speech and Audio Signal Processing, John Wiley & Sons, Inc.
5. An Introduction to Biometrics, biometric consortium,
<http://www.biometrics.org/html/introduction.html>
6. Biometrics Institute (The biometric resource center), "Product available for voice recognition", <http://www.biomet.org/voiceproducts.html>
7. BiOLD, "BiOLD products", <http://www.bioid.com/products/products.html>
8. T-NETIX, "SpeakEZ Voice Print speaker verification",
<http://www.t-netix.com/SpeakEZ/default.asp>
9. VeriVoice, "VeriVoice products",
<http://www.verivoice.com/products.htm>
10. Veritel Corporation, "VoiceCheck software development kit",
<http://www.veritelcorp.com/Products/sdk.html>
11. Keyware Technologies, "CAS Guard",
<http://www.keyware.com/techsol/pages/casguard.asp>
12. Veritel Corporation, "Speak N Set",
<http://www.veritelcorp.com/Products/speaknset.html>
13. Rabiner, Lawrence & Juang, Biing-Hwang. Fundamentals of Speech Recognition. New Jersey: Prentice Hall, 1993.
14. Mohamed F. BenZeghiba, Herve Bourlard, 2001, User Customized HMM/ANN-Based Speaker Verification, IDIAP-RR 01-32

-
15. PravinkumarPremakanthan, Wasfy. B. Mikhael, 2001, Speaker verification/recognition and the importance of selective feature extraction: Review, *Circuits and Systems, MWSCAS Proceedings of the 44th IEEE 2001 Midwest Symposium on*, Vol. 1, 57- 61
 16. D. Hanselman, B. Littlefield (1998), Mastering MATLAB 5, Prentice Hall, International, Inc.
 17. MATLAB official site
<http://www.mathworks.com/>
 18. Illustration of the LBG Vector Quantization algorithm
http://lcavwww.epfl.ch/~minhdo/asr_project/vqlbg_images/index.html

APPENDIX A : GUI & MANUAL

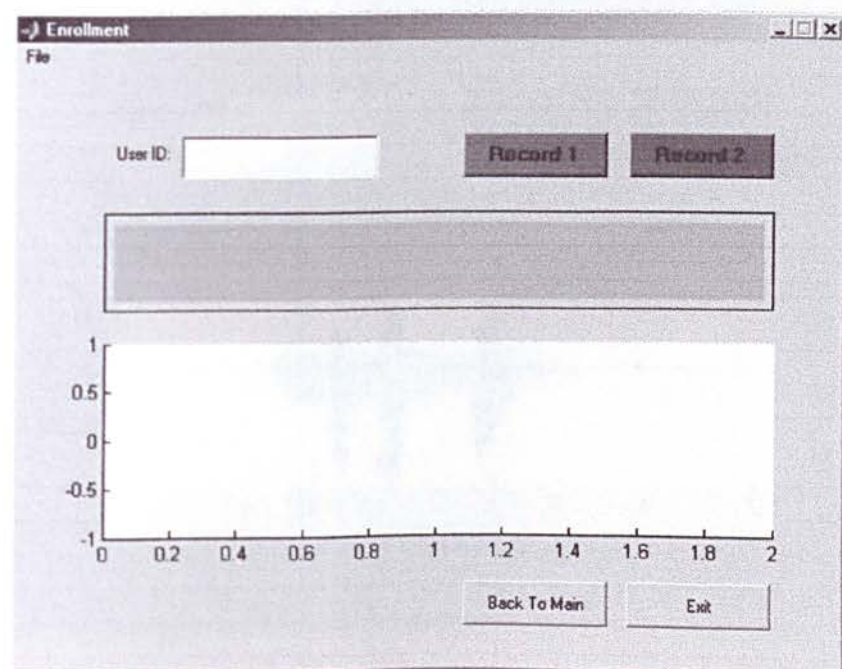
1. Open Matlab and type asv/ASV. The main menu of the system will appear.



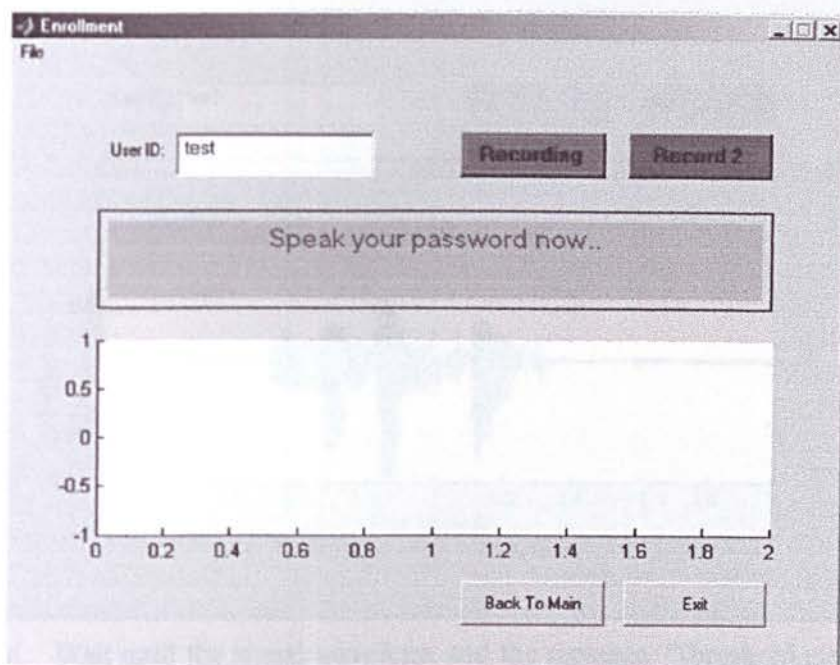
2. Choose which options from the two available module.

2.1 Enrollment Module

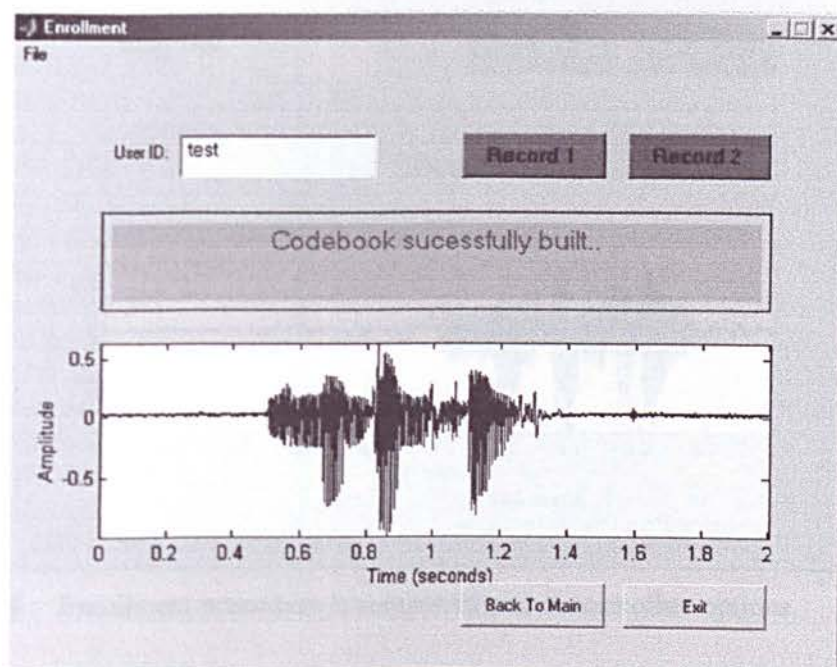
- a. The enrollment screen will be brought up.



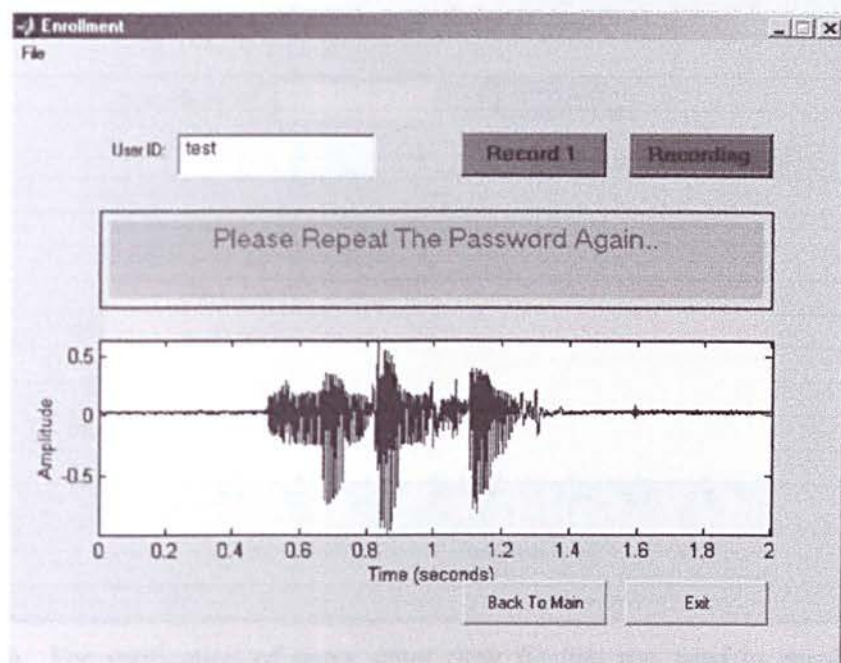
- b. For enrollment of new users, enter an ID and then click the Record 1 button. Wait until the message “Speak your password..” appears before speaking into the microphone. *Note: The duration of recording is 2 seconds.



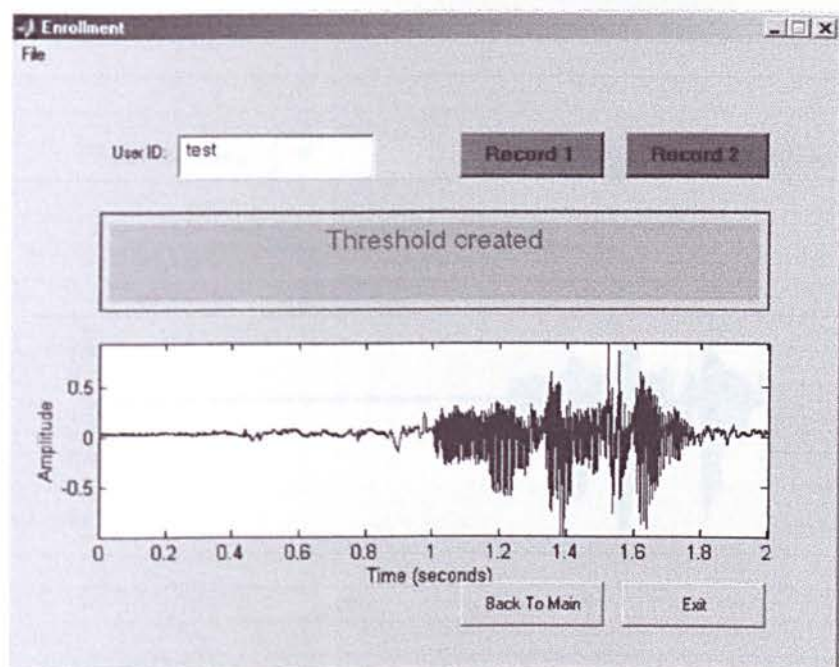
- c. The system will perform the necessary operations (It may take awhile) and then the signal waveform will be plotted and the message “Codebook sucessfully built.. “ will be displayed.



- d. Click the Record 2 button and wait until the message “Please repeat the password again..” appears before speaking into the microphone.



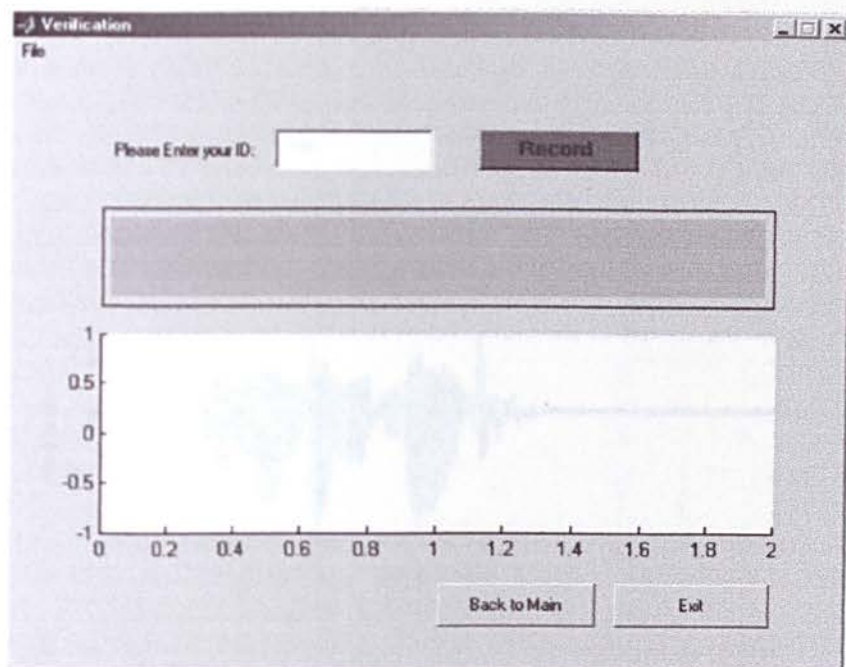
- e. Wait until the signal waveform and the message “Threshold created” appears.



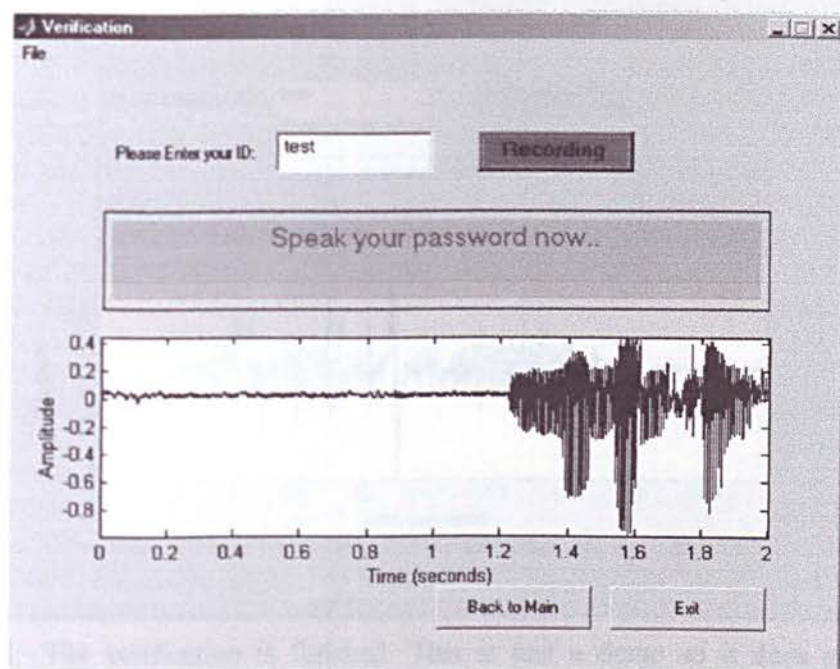
- f. Enrollment procedure is completed and choose other options.

2.2 Verification Module

- a. The verification screen will be brought up.

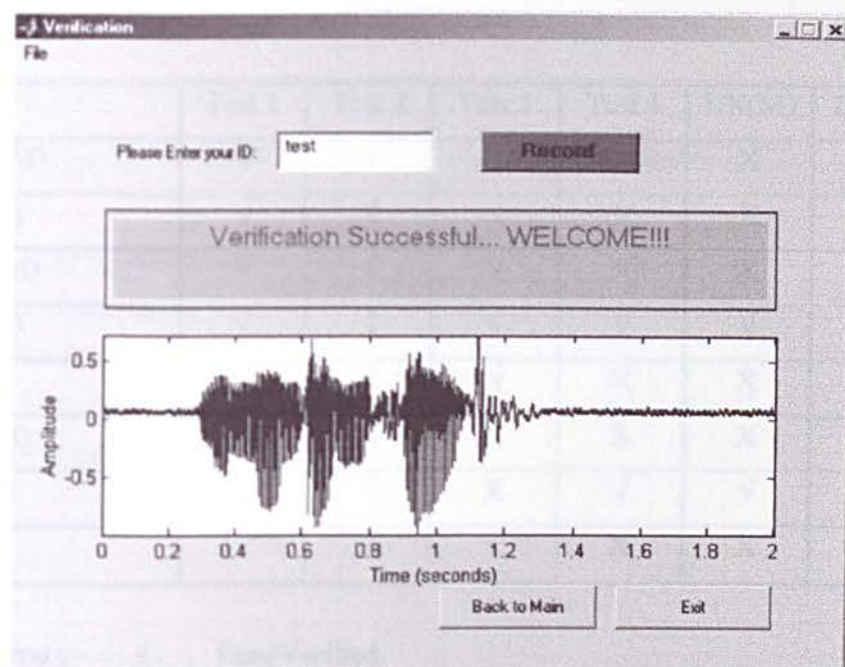


- b. For verification of users, enter their ID that was used in enrollment and then click the Record button. Wait until the message "Speak your password now.." appears before speaking into the microphone.

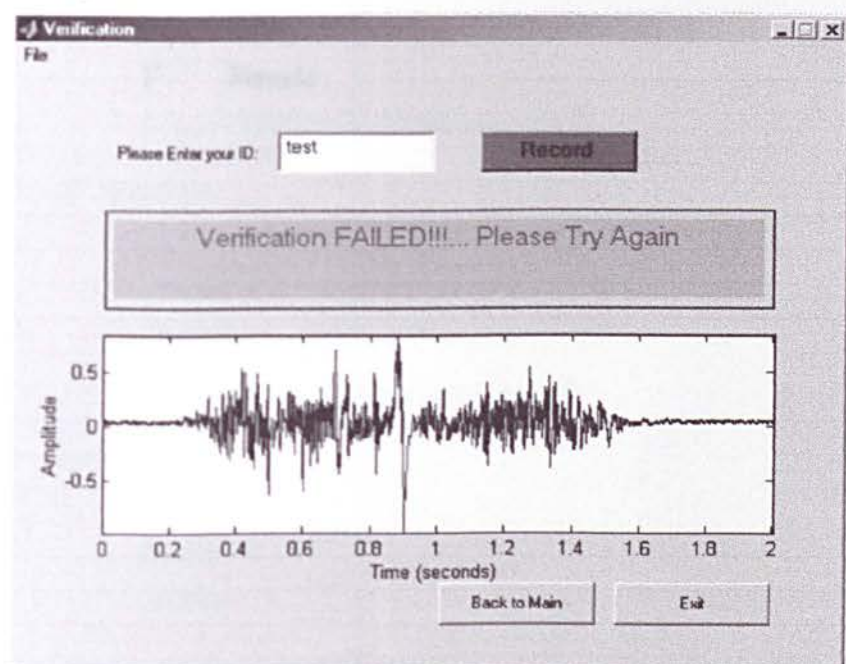


- c. The system will perform the necessary verification operation. If the verification is successful the signal waveform will be plotted and the

message "Verification Successful... WELCOME!!! " will be displayed.



Otherwise the message will display "Verification failed... Please try again.."



d. The verification is finished. This is just a demo so it does not do anything but testing the verification function.

APPENDIX B : TEST RESULTS

	Test 1	Test 2	Test 3	Test 4	UX(M)	UX(F)
Shahman (M)	√	√	X	√	X	X
Samsul (M)	√	X	√	X	X	X
Syahiran (M)	√	√	√	X	X	X
Safwan (M)	√	√	√	√	√	X
Brett (M)	√	√	√	X	X	X
Stephen (M)	√	X	X	X	X	X
Faridah (F)	√	√	X	√	√	X
Bojana (F)	√	√	√	X	X	√

Legend :

- √ Pass/Verified
- X Fail/Rejected
- UX Unknown speaker
- M Male
- F Female

APPENDIX C : CALL STRUCTURES

The following is the hierarchy of the main files that are called in the system. Some files are listed twice because they are called from different places. Data files have the extension *.mat*.

Graphical User Interface

MainMenu.fig, mainmenu.m
 Enrollment.fig, enrollment.m
 Verification.fig, verification.m
 TestMic.fig, testmic.m

Core Functions

mainmenu.m
 enrollment.m
 id_train_speaker.m
 locatespeech.m
 melcepst.m
 vqlbg.m
 Create UserID*.mat (*Codebook created*)
 id_treshold.m
 locatespeech.m
 melcepst.m
 disteu.m
 Update UserID*.mat (*threshold added*)
 verification.m
 locatespeech.m
 melcepst.m
 disteu.m

*UserID is the ID that the user entered at the enrollment screen.